Research Initiation Award: Adaptive Coding on Nonstationary Communication Channels with Feedback

National Science Foundation Grant NCR-9009877*

September 15, 1990 - June 30, 1993

FINAL REPORT

Professor Stephen B. Wicker
Principal Investigator

Coding and Information Theory Laboratory
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332 USA
(404) 894-3129
wicker@ee.gatech.edu

*Georgia Tech OCA Number E21-F87
1. Introduction

For more than a decade following its genesis in 1949, the field of error control coding focused on the problem of error control for unidirectional communication channels. This design problem may be stated as follows: how can one encode a block of data so that the receiver can correct a sufficient number of the error patterns induced by the channel while simultaneously maximizing the data rate of the system? Error control coding for unidirectional channels is frequently called Forward Error Correction and is referred to here by its acronym "FEC". The advent of the communication network led to the development of a second approach to error control. Networks frequently provide for bidirectional communication between users, allowing one user to react to a message from another user by sending a response of some sort. The return channel that carries the receiving user's response is frequently called the feedback channel. The error control system design problem can now be augmented by the following question: how can one use the feedback channel to improve the receiver's ability to correct errors while further improving the data rate of the communication system? Error control coding systems that take advantage of a feedback channel are referred to here as bidirectional error control (BEC) systems.

This report summarizes the results of a research program that examined various means of constructing BEC systems. It is intended to illustrate the publications that emerged from this effort. The publications are listed at the end of the report, while reprints of the most important publications are included as appendices.

2. Single Decoder Type-I Hybrid-ARQ Protocols

The simplest and oldest of the BEC systems introduce redundant symbols into the data stream solely for the purpose of error detection. The redundant symbols ensure that there are a large number of sequences of symbol values that are invalid. If the receiver sees an invalid pattern in a demodulated packet of symbols, it requests a retransmission of that packet under the natural assumption that one or more error have been caused by noise on the channel. These "detection only" BEC schemes are called "Automatic Repeat Request" (ARQ) protocols [LiN]. They provide significantly better reliability performance than FEC systems that introduce the same number of redundant symbols. The price for this improved reliability performance is exacted in the form of a reduction in throughput caused by retransmission requests. If the channel remains good and received symbol errors remain infrequent, the ARQ protocol performs quite well. But, if the channel degrades to the point that one or more symbol errors occur in every received packet, then the ARQ protocol fails catastrophically.
In applications where symbol errors occur far too frequently for the use of an ARQ protocol, the data can be encoded for both error detection and error correction. The error control system is designed to correct the most frequently occurring error patterns while reserving detection capacity for the less frequently occurring patterns. The throughput performance is improved through the correction of the frequently occurring patterns while reliability performance is maintained by not trying to correct the less frequently occurring (and therefore higher weight and more likely to cause a decoder error) error patterns. Such a combined ARQ/FEC protocol is called a "type-I hybrid-ARQ" protocol [LIN].

Type-I hybrid-ARQ protocols have traditionally been implemented using two encoders and two decoders (FEC error correction combined with CRC error detection). One of the first research projects pursued under this contract was the investigation of single encoder/single decoder type-I systems. This work was motivated by a paper by Drukarev and Costello [DRU] in which a sequential FEC decoder for convolutional codes is modified to implement two different type-I hybrid-ARQ protocols. In each of the two protocols, the decoder computes a reliability statistic using information generated during the execution of a sequential decoding operation. The reliability statistic is compared to a threshold to determine whether the decoded data obtained from a received packet is sufficiently reliable. If not, a retransmission of the packet is requested. The retransmission threshold is used to establish a balance between error detection and error correction within the decoder. If detection is emphasized, reliability is favored at the expense of throughput. If correction is emphasized, the opposite effect is seen. Single encoder/single decoder protocols have several advantages over double encoder/double decoder systems. The most obvious is that there is no longer any need for the additional CRC encoder and decoder circuitry. Though CRC circuitry is very simple, it can introduce a substantial amount of delay. The single encoder/single decoder approach also allows for the dynamic reallocation of error correction and error detection capacity through the manipulation of the retransmission threshold. The FEC/CRC approach does, however, allow for more flexibility in setting the error detection parameters without affecting the error correction system. The Principal Investigator and his students have applied the single encoder/single decoder approach to a series of decoding algorithms.

2.1 Majority Logic Decoders for Cyclic and Convolutional Codes

The first of the FEC decoder modifications investigated focused on majority logic decoders for convolutional [5] and cyclic block [7],[16] codes. Majority logic decoders poll a set of check sums computed from a received codeword to estimate the values of the error bits affecting individual information bits. The extent of the majority formed during these polling operations is a source
of reliability information that can be used in the definition of a type-I protocol.

The hybrid-ARQ majority-logic decoding rule is as follows. Let $\eta$ be the number of the $J$ check sums orthogonal on error bit $e$ that have values of one. Let $\tau$ be a nonnegative integer less than $J$. If $\eta \geq \lfloor J/2 \rfloor + \lceil \tau/2 \rceil + 1$, then $e$ is assumed to have a value of one. If $\eta \leq \lfloor J/2 \rfloor - \lceil \tau/2 \rceil$, then $e$ is assumed to have a value of zero. If $\lfloor J/2 \rfloor - \lceil \tau/2 \rceil < \eta < \lfloor J/2 \rfloor + \lceil \tau/2 \rceil + 1$ then a retransmission of the packet is requested.

Figures 1 and 2 show the impact on reliability and throughput for a typical case. Note that as the width of the retransmission region increases, the reliability performance improves with respect to the FEC case. Note also the penalty paid in the form of reduced throughput. In all cases, however, there is an operating region that allows for good throughput performance while providing significantly improved reliability performance. Analysis and simulation results are discussed in detail in [5], [7], and [16].

2.2 Reed-Solomon and BCH Decoders

Consider a bounded distance FEC decoder for a linear block code. Under most channel conditions, if the decoder corrects a large number of errors in a received packet, it is more likely to be committing a decoder error than if it corrects few or no errors. A type-I protocol can be created by simply establishing an allowed error correction threshold somewhere below the maximum correction capability of the code. If the number of errors corrected in a received word exceeds the threshold, then a retransmission request is generated. This simple idea leads to the development of an extremely powerful type-I protocol based on Reed-Solomon codes. In the Berlekamp-Massey and Euclidean decoding algorithms for Reed-Solomon and BCH codes, the degree of the error location polynomial indicates the number of errors to be corrected by the decoder. In the RS-HARQ system [4] the degree of the error location polynomial is compared to a retransmission threshold and the appropriate action taken. The geometry of the Reed-Solomon codes make this an unexpectedly powerful system. Reed-Solomon codes are excellent error correcting codes, but the decoding spheres defined by a Reed-Solomon bounded distance decoder do not define a good sphere packing. This can be seen by noting that the probability of decoding failure in most Reed-Solomon error control systems is substantially higher than the probability of decoder error. The space between the decoding spheres can thus be used to great advantage for error detection in a type-I hybrid-ARQ protocol.

In [9], [19], and [21] the RS-HARQ system is extended to include erasure decoding and is analyzed for the case of the slowly fading Rayleigh channel. In the extended system, the number of erasures $s$ and the number of errors $e$
to be corrected in a received packet are used to compute a reliability statistic

\[ l = (2e + s) \].

\( l \) is compared to the effective diameter \( d_e \) of the RS-HARQ system to determine whether a retransmission request is to be generated. Figures 3 and 4 indicate the performance provided for a slowly fading Rayleigh channel.

2.3 The Viterbi Decoder and the Error Trapping Algorithm

Yamamoto and Itoh were the first to introduce a type-I hybrid-ARQ protocol based on the Viterbi decoder [YAM]. They noted that the Viterbi decoder is more likely to make a decoder error whenever the partial path metric of the maximum likelihood path is close to that of a nonsurviving path at one or more of the nodal decision points. The Principal Investigator and one of his students (B. A. Harvey, doctoral thesis [2]) developed an alternative type-I protocol by taking a somewhat different approach. It was noted that the Viterbi decoder is more likely to make a decoder error whenever the partial path metric of the maximum likelihood path increases rapidly over a short span of the trellis. This observation lead to the development of the error trapping algorithm [6], [17]. The rate of increase of the partial path metric for all paths is computed over a sliding window of fixed length. Any path whose rate of increase exceeds a preset threshold is declared unreliable. If all paths are declared unreliable before decoding is completed, then a retransmission request is generated.

The error trapping algorithm has a number of interesting properties when applied to hard decision Viterbi decoders. Using a combinatorial attack, it is possible to derive an exact expression for the probability that a packet is correctly accepted on any given transmission. No such expression has been found for the Yamamoto and Itoh algorithm. The error trapping algorithm also has a very natural implementation in Viterbi decoders that use the traceback algorithm to make decisions on information bits.

2.4 Other Modified FEC Systems

In [3] the application of the Yamamoto and Itoh algorithm to trellis coded modulation systems is investigated. Upper and lower bounds on throughput and reliability are derived; the bounds indicate that the performance of the resulting type-I system is significantly better than that of the TCM FEC system.

The most recent efforts in the area of modified FEC systems for type-I protocols focused on Reed-Muller codes. Reed-Muller codes do not currently receive a great deal of attention because they are "weak codes that are easy to decode" [BERL]. As there are now several strong codes that are easy to decode, Reed-Muller codes have lost the initial appeal that sent them out into the far
reaches of the solar system aboard the Mariner spacecraft of the late 1960's and early 1970's. Increased attention is expected, however, as the extraordinarily fast RM decoders find applications in optical systems with high data rates.

It was found that significant throughput of reliable data can be obtained from a hypothetical Mariner spacecraft with a severely reduced transmitter power level through retransmission requests and type-I hybrid-ARQ Reed-Muller decoding. The algorithm presented in [12] and [26] is a modification of Green's maximum likelihood FEC decoding algorithm for first-order Reed-Muller codes (also known as the Green machine). The best results were obtained, however, through the creation of a type-II hybrid-ARQ Reed-Muller protocol. Type-II protocols provide a simple form of packet combining, the subject of the next section.

3. Packet Combining

Type-I hybrid-ARQ protocols postpone the catastrophic failure exhibited by ARQ protocols as the communication channel degrades, but they are unable to avoid it altogether. Eventually a point is reached when uncorrectable error patterns are occurring with such frequency that throughput on the type-I system becomes negligible. It is possible to further extend the operating range of a BEC system in the low SNR region through the use of packet combining. Packet combining systems differ from type-I protocols in that they do not discard received packets that have caused the generation of a retransmission request, but instead combine all received packets in an attempt to create a single packet that can be reliably decoded. As the channel deteriorates, the packet combining system uses an increasing number of packets, maintaining at least a small level of throughput under extremely adverse channel conditions.

Packet combining systems can be divided into two categories: diversity combining systems and code combining systems. Diversity combining systems use symbol-by-symbol combining to create a single rate R encoded packet from several copies of a rate R encoded packet. The goal is to increase the effective signal to noise ratio of the symbols comprising the packet resulting from the combining operation. In code combining systems [CHASE], L received codewords encoded at rate R are combined to create a single noise corrupted codeword with rate R/L. The goal is to decrease the rate of the code until there is sufficient redundancy in the packet resulting from the combining operation to reliably correct the errors caused by the noise on the channel.

3.1 Convolutional Codes: Majority Logic Diversity Combining

Majority logic diversity combiners poll multiple copies of a received symbol, selecting the majority value for the corresponding coordinate position in the
combined packet. In [5] a hard decision voting scheme was used to implement majority logic combining in the type-I hybrid-ARQ majority logic decoder for convolutional codes discussed in section 2.1. To ensure that no ties occur, retransmissions in this scheme alternate between packets containing two copies of the information bits and packets containing two copies of the parity bits (assuming a rate 1/2 code is in use; other strategies can be used for other rates). When combined with the packet from the initial transmission, the retransmitted packets ensure that there is always an odd number of copies of both the information and parity bits. Figure 5 shows the resulting improvement in throughput performance at low signal to noise ratios. Note that the throughput reduction with decreasing SNR is much more graceful than in the case without combining (Figure 2).

Majority logic combining can also be applied to hard decision Viterbi decoders that have been modified to implement the error trapping algorithm discussed earlier [22]. In this case the retransmitted packet formatting strategy is independent of the code rate. Retransmissions alternate between packets containing two copies of the even numbered bits and packets containing two copies of the odd numbered bits. Figure 6 shows the resulting improvement in throughput performance.

3.2 Convolutional Codes: Averaged Diversity Combining

Diversity combining can be extended to soft decision Viterbi decoders through the use of symbol copy averaging. In averaged diversity combining systems, the demodulated values for all received copies of a given bit are simply averaged by adding them up and dividing the result by the number of copies used in the summation. In [14] and [22] averaged diversity combining is applied to soft decision Viterbi decoders that have been modified to implement the Yamamoto and Itoh type-I hybrid-ARQ algorithm. The resulting improvement in performance is shown in Figure 7. [14] and [22] also show that the averaged diversity combining system provides exactly the same reliability and throughput performance as a code combining system that interleaves all received packets to create convolutional codewords of lower rate.

In [14] and [22] the averaged diversity combining results are extended to cover "moderately varying channels" (i.e. channels that are constant over a packet transmission time, but may vary from packet to packet). One of the more interesting results to emerge from this study was that averaged diversity combiners perform as well as weighted diversity combiners over many highly nonstationary channels.
3.3 Convolutional Codes: Weighted Diversity Combining

For moderately varying channels, it is sometimes necessary to weight the received packets before combining to prevent a single extremely noisy packet from forcing a large number of retransmission requests. In [CHASE] a series of weights are derived for code combining over Binary Symmetric and AWGN channels. These weighting factors are obtained using ideal channel side information. In [14] and [22] these weights are used to implement a weighted diversity combining scheme for hard and soft decision Viterbi decoders. [14] concludes by demonstrating a method for deriving the packet weights using side information generated by the Viterbi decoder, obviating the need for some type of channel noise measurement. It is shown that the resulting performance degradation caused by the suboptimal weights is extremely small.

3.4 Code Combining System Based on Punctured MDS Codes

Code combining systems use multiple received packets to create noise corrupted codewords from codes with increasingly lower rates. From an implementational standpoint this creates a problem, for the code combining decoder must be capable of decoding several different rates. In this project an effort was made to identify codes whose structure allows for a natural (and therefore easily implemented) approach to variable rate decoding. In [10] and [23] this approach is applied to create a code combining system based on punctured MDS codes (the MDS codes include the Reed-Solomon codes). It is shown that codewords from an extremely low rate, long MDS code can be partitioned to form a series of codewords from high rate punctured MDS codes. Codewords from this series can be combined to create increasingly lower rate MDS codewords down to the limiting rate of the "mother code". The decoder for the mother code can be used to decode all of the smaller codewords. This system uses the type-I protocol discussed earlier to generate retransmission requests.

In [10] and [23] this system is analyzed in detail for the case of Reed-Solomon codes in a type-II hybrid-ARQ protocol (a type-II protocol is a code combining system in which combining operations are limited to two packets at a time). An exact method for characterizing packets that have caused the generation of a retransmission requests is developed and used in a graph theoretic analysis to obtain exact throughput and reliability expressions.

3.5 The Reed-Muller Type-II Hybrid-ARQ Protocol

The type-I hybrid-ARQ Reed-Muller protocol discussed earlier can be extended to create a type-II protocol through puncturing and the use of a series of RM subcodes [12] and [26]. The modification allows for moderate levels of throughput at extremely low signal to noise ratios, providing some
improvement over the performance of the type-I protocol. Unfortunately the relatively poor performance of long Reed-Muller codes prevented the definition of a practical RM system that combines more than two packets.

4. Rate Switching Systems

Packet combining systems can adapt rapidly to changes in channel conditions, but are limited in the number and selection of effective code rates that they can use. Typically all of the code rates available are of the form R/L, where R is a base rate and L is an integer. If the channel is sufficiently slowly varying with respect to the data rate (e.g. satellite links vary slowly with the weather), a different approach can be taken. It is possible to adopt a rate switching system whose code rate is selected through a channel state estimation procedure. The rate switching systems can provide theoretically infinite rate resolution. In practice some concessions must be made in the selection of code rates, but the possibilities are still more numerous than in the case of packet combining systems.

It has been noted by a number of authors that the retransmission statistics exhibited by a hybrid-ARQ protocol can be used as the basis for an adaptive error control system. This idea is pursued in [8], where a channel estimation strategy based on the number of retransmissions within a frame of packets is introduced and analyzed. The channel is modeled as an M-state Markov chain. Each state in the chain has a corresponding type-I protocol that provides acceptable performance as defined by a set of characteristic functions.

The rate switching approach is further developed in [1] and [13], where a sequential testing scheme based on retransmission statistics is used to drive a rate switching system. The sequential test is able to react much more quickly to changes in the channel noise level than the framed approach in [8]. The reliability and throughput performance of the sequential system is shown to be quite close to that of an ideal rate switching system with perfect channel state information.

In [1] and [13] several rate switching systems are developed and analyzed. The most promising systems use rate compatible punctured convolutional (RCPC) codes or Reed-Solomon codes in type-I hybrid-ARQ protocols. The RCPC system in particular provides for an extremely simple implementation in which the encoder and decoder need not be substantially modified beyond the type-I hybrid-ARQ modification.
Technical Publications and Presentations: NCR-9009877

Doctoral Theses


Books


Refereed Journal Articles


Refereed Conference Presentations


Referenced Work by Other Authors


Figure 1: Reliability Performance for (2,1,55) Convolutional Code in a Majority-Logic Type-I Hybrid-ARQ Decoder (Packet Length = 1000).
Figure 2: Throughput Performance for (2,1,55) Convolutional Code in a Majority-Logic Type-I Hybrid-ARQ Decoder (Packet Length = 1000).
Figure 3: Word Error Rate for a (64, 56) Reed-Solomon Code in the RS/HARQ System with erasures.

Figure 4: Throughput for a (64, 56) Reed-Solomon Code in the RS/HARQ System with erasures.
Figure 5: The lower bound on throughput and simulation results for a (2,1,55) convolutional code in a diversity combining majority-logic hybrid-ARQ protocol. Packet Length = 2000
Figure 6: Throughput Performance for Hard Decision Viterbi Decoding with and without Majority-Logic Diversity Combining

(Decoder uses Error Trapping Algorithm with a (2, 1, 3) Convolutional Code)
Figure 7: Throughput Performance for Soft Decision Viterbi Decoding with and without Averaged Diversity Combining

(Decoder uses Yamamoto-Itoh Algorithm with a (2, 1, 3) Convolutional Code)
APPENDICES

Reprints of several papers are included as appendices.
Reed–Solomon Error Control Coding for Rayleigh Fading Channels with Feedback

Stephen B. Wicker, Member, IEEE

Abstract—The use of nonbinary block error control codes over Rayleigh fading channels with feedback is examined. It is assumed that the fading is slow with respect to the rate of symbol transmission. Expressions are derived for the probabilities of channel symbol error and erasure, which are in turn used to develop expressions for code symbol error and erasure. Two erasure generation mechanisms are considered, one based on the existence of channel amplitude side information, the other not. This analytical framework is used to evaluate the performance of the Reed–Solomon/hybrid-ARQ protocol (RS/HARQ) over fading channels with feedback. The RS/HARQ system uses erasure decoding in a hybrid-ARQ protocol to provide excellent reliability performance at the expense of a reduction in throughput. The RS/HARQ protocol allows for the variation of the erasure threshold and the effective diameter of the decoding operation, providing a powerful, flexible error control system for a variety of fading channel applications.

I. INTRODUCTION

In a large number of communication networks, the links established between users are bidirectional. The existence of two channels (forward and feedback when referenced to a single user) offers a significant opportunity for the development of powerful automatic-repeat-request (ARQ) error control systems. It has been shown in the literature that ARQ systems and their hybrids can provide better reliability performance than their forward-error-correcting (FEC) counterparts at the expense of a reduction in throughput [1]–[3]. A block diagram for a generalized ARQ system is shown in Fig. 1. The underlying functional principle of ARQ systems is the use of the feedback channel by the receiver to request retransmissions of data packets that are believed to be unreliable. In this paper the case of the Rayleigh fading forward channel with additive white Gaussian noise is considered. The feedback channel is assumed to be ideal (constant amplitude and noise-free).

In the next section the performance of nonbinary block codes is considered in conjunction with both channel symbol and code symbol interleaving over Rayleigh fading channels. Two erasure generation mechanisms are considered in this development, one assuming the existence of ideal channel amplitude side information and the other not. Using both approaches, code symbol error and erasure processes are characterized in terms of the channel symbol error and erasure processes. It is shown that code symbol interleaving provides better symbol error and erasure performance than channel symbol interleaving.

In the following section, a Reed–Solomon hybrid-ARQ (RS/HARQ) error control system is described in detail. This RS/HARQ system is an extension of the system described in an earlier paper [1]. The new system allows for the use of erasure decoding, providing a significant performance improvement when used in applications involving fading channels.

In the final section, examples are used to illustrate various aspects of the performance of the RS/HARQ system over fading channels with feedback.

II. THE PERFORMANCE OF NONBINARY CODES OVER FADING CHANNELS

The amplitude of the forward channel is defined as a Rayleigh random variable \( a \) with the probability density function

\[
p_a(a) = 2ae^{-a^2}.
\]  

(1)

The forward channel is also corrupted by additive white Gaussian noise (AWGN) with one-sided power spectral density \( N_0 \). It is assumed throughout the rest of this paper that

Fig. 1. Block diagram for a generalized ARQ error control system.
channel phase variations due to multipath fading are detected and removed during the demodulation process. Methods for realizing such performance through pilot tone techniques are discussed in [4] and [5]. It is also assumed that channel fading is frequency-nonselective.

To determine the performance of nonbinary codes over this channel, one must first select a modulation format and derive expressions for the probabilities of channel symbol error and erasure. There are two basic methods for generating channel symbol erasures that are distinguished by the existence or nonexistence of forward channel amplitude side information at the receiver; both are considered in the following analysis.

Let the channel symbol alphabet have cardinality $2^b$ and the code symbol alphabet have cardinality $2^m$. Nonbinary codes provide a level of burst error correction for fading channels that is a function of the amount by which $m$ exceeds $b$ (it is assumed that the symbol transmission rate is greater than the fade rate). The rationale is that a deep fade may affect several consecutive channel symbols while only affecting a few code symbols, thus allowing for the use of higher rate codes. It follows that this burst error correcting capability is lost if the channel symbols are individually interleaved (as opposed to their being interleaved in clusters, each cluster corresponding to a code symbol).

The channel symbol error and erasure expressions developed in this section are translated into code symbol error and erasure expressions for both the code symbol and channel symbol interleaved channels. An example is provided to demonstrate that nonbinary codes provide better performance when used in conjunction with code symbol interleaving as opposed to channel symbol interleaving.

A. Erasure Generation Without Side Information

If no channel amplitude information is available, the modem signal space is partitioned into several nonerasure (reliable) decision regions and an erasure (unreliable) decision region. The declaration of erasures by the receiver is then solely a function of the received symbol energy. Consider the case of the 8-PSK modulation format [6]. Fig. 2 shows how the signal space is partitioned into eight nonerasure decision regions $\{A_0, A_1, \ldots, A_7\}$ and an erasure region $A_*$. In the general case an $n$-dimensional signal space $\Lambda$ with $(2^n + 1)$ decision regions $\{A_0, A_1, \ldots, A_{2^n-1}\}$ is considered. Suppose that the channel symbol corresponding to the decision region $A_i$ is transmitted. A conditional probability density function $p(z|A_i, A_i)$ is derived for a fixed channel amplitude $a$ and a received signal $z = (z_0, z_1, \ldots, z_{n-1})$. This conditional pdf can be obtained by inverting the product of the characteristic functions for the Gaussian and Rayleigh processes defining the channel (e.g. the MPSK case is treated by Proakis in [7, appendix 7A]). The probability of channel symbol error as a function of channel amplitude is then

$$p_{ce}(a) = E_i \left\{ \sum_{j \neq i} \left( \int_{\Lambda_j} p(z|A_i, A_j) \, dz \right) \right\}$$

where the expected value operation is taken over all possible transmitted signals $i$ corresponding to reliable decision regions $\Lambda_i$. The probability of channel symbol erasure as a function of $a$ is

$$p_{ca}(a) = E_i \left\{ \int_{\Lambda_j} p(z|A_i, \Lambda_j) \, dz \right\}.$$  

It is important to note that when side information is unavailable, erasures can be caused by the AWGN process as well as the channel fading process.

B. Erasure Generation with Side Information

If it is assumed that side information containing the exact value of the channel amplitude $a$ is available, a simpler approach to erasure generation can be considered. Hagenauer and Lutz [8] have examined the case in which erasure generation is based solely on the value of $a$, eliminating the impact of the AWGN process on the probability of channel and code symbol erasures. Let $\lambda_*$ be the erasure threshold. A received channel symbol shall be declared an erasure any time the channel amplitude is less than $\lambda_*$. The probability of this occurring is

$$p_{ca}^{SI} = \int_0^{\lambda_*} p_a(a) \, da.$$  

The derivation of an expression for the probability of channel symbol error is also quite simple. Once the channel amplitude is known for a received channel symbol and it has been determined that the amplitude is not below the erasure threshold, the decision regions for the demodulator are scaled to match the channel amplitude and the symbol decision is made. The 8-PSK decision regions for the side information case are shown in Fig. 3. The signal scaling caused by the fading channel amplitude is indicated. Due to the radial symmetry of the signal constellation in this example, the decision regions do not change with the value of $a$. This is not the case, however, with nonconstant envelope modulation formats (e.g., ASK and QAM). The probability of channel symbol error for channel amplitude $a$ and a given modulation format is obtained by taking the standard AWGN symbol error rate expression and weighting the signal energy by $a^2$. For example, the probability of bit error for coherent BPSK, which is used extensively in
A received code word, but not that between adjacent channel symbols comprising a single received code symbol.

A code symbol erasure occurs whenever one or more of the \( m/b \) constituent channel symbols are declared to be erasures. The derivation of the code symbol erasure probabilities for both channel symbol erasure generation mechanisms is straightforward. For the case where side information is not available and the channel amplitude is a constant \( a \), the probability of there being at least one erased channel symbol among \( m/b \) channel symbols is

\[
p_s(a) = 1 - (1 - p_{cs}(a))^{m/b}.
\]  

(6)

The code symbol erasure probability \( p_s \) is then obtained through an expected value operation using the probability density function for \( a \). For the case with side information, the probability of code symbol erasure is determined solely by the value of \( a \), which is constant during the transmission of the code symbol. The probability of symbol erasure in this case is simply the probability that the value of \( a \) at any given moment is below the erasure threshold \( \lambda_s \). The following expressions result (see (7) below):

A code symbol error occurs whenever the code symbol has not been declared an erasure and one or more of the constituent channel symbols is in error. For the case when side information is not available, the code symbol error probability for constant channel amplitude \( a \) is

\[
p_e(a) = (1 - p_{cs}(a))^{m/b} - (1 - p_{ce}(a) - p_{cs}(a))^{m/b}.
\]  

(8)

The code symbol error probability is then obtained through an expected value operation. The corresponding result for the case with side information is similar, though the limits of integration rule out the possibility of a channel symbol erasure, simplifying the integrand. The probability of code symbol error in both cases is (see (9) below):

C. Code Symbol Errors and Erasures on Code Symbol Interleaved Slowly Fading Channels

In the literature frequent reference is made to “fast” and “slowly” fading channels. These terms can be understood only when taken in relation to something; for example, the channel symbol transmission rate. It is assumed here that a slowly fading channel exhibits fades whose duration exceeds the time required to transmit several channel symbols. A fade affecting one channel symbol is thus highly likely to affect temporally adjacent channel symbols. The techniques used to combat this correlative effect are one of the concerns of this paper.

Two interleaving techniques are considered: interleaving at the channel symbol level and interleaving at the code symbol level.

The analysis of the code symbol interleaved slowly fading channel is predicated on the following assumptions.

- Channel amplitude is constant over the \( m/b \) consecutive channel symbols constituting a code symbol.
- The channel symbols are interleaved to infinite depth in clusters corresponding to individual code symbols. The noise processes affecting adjacent code symbols within a code word are thus uncorrelated.

In this case interleaving is being used to eliminate the correlation of the noise/fading process affecting adjacent symbols in the examples in a later section, is

\[
p_{csI}(a) = \frac{1}{2} \text{erfc} \left( a \sqrt{\frac{R E_b}{N_0}} \right)
\]  

(5)

where \( R \) is the rate of the code in use.

D. Code Symbol Errors and Erasures on Channel Symbol Interleaved Slowly Fading Channels

The analytical ground rules are changed considerably for the case of channel symbol interleaving. The following assumptions are made.

- The channel amplitude is constant over the time required to transmit one or more channel symbols.

\[
p_s = \left\{ \begin{array}{ll}
\int_0^a \left[ 1 - (1 - p_{cs}(a))^{m/b} \right] p_a(a) \, da, & \text{no side information} \\
\int_{\lambda_s}^\infty \left[ (1 - p_{cs}(a))^{m/b} - (1 - p_{ce}(a) - p_{cs}(a))^{m/b} \right] p_a(a) \, da, & \text{side information}
\end{array} \right.
\]  

(7)

\[
p_e = \left\{ \begin{array}{ll}
\int_0^a \left[ (1 - p_{cs}(a))^{m/b} - (1 - p_{ce}(a) - p_{cs}(a))^{m/b} \right] p_a(a) \, da, & \text{no side information} \\
\int_{\lambda_s}^\infty \left[ 1 - p_{csI}(a) \right]^{m/b} p_a(a) \, da, & \text{side information}
\end{array} \right.
\]  

(8)
The channel symbols are interleaved to infinite depth. The noise processes affecting adjacent channel symbols are thus uncorrelated.

From the decoder's perspective, the channel symbol interleaved channel appears to fade more rapidly than the code symbol interleaved channel, which in turn fades more rapidly than a stationary channel. It should be noted, however, that for both interleaved channels, the physical propagation medium may be the same: a Rayleigh fading channel whose fade duration exceeds the time required for the transmission of several channel symbols. It is the method used to format the coded information prior to transmission that differs between the code symbol and channel symbol interleaved cases. Consider, for example, a mobile radio channel that is used for the transmission of data. Variations in vehicle velocity cause the physical channel fade frequency and duration to vary. In almost all applications, however, the fading is slow with respect to the channel symbol transmission rate. Assuming sufficient depth, channel (code) symbol interleaving ensures that a given fade does not affect adjacent channel (code) symbols. The effective fade rate seen by the receiver after deinterleaving thus appears to be faster than the channel (code) symbol transmission rate.

The expected value operations of the previous section are performed at the channel symbol level instead of the code symbol level for the case of the channel symbol interleaved channel. Consider first the probability of channel symbol erasure for the case without side information. The expected value of the channel erasure probability is computed using the probability density function for a as follows:

\[ P_e = \int_0^\infty p_e(a) p_a(a) \, da. \]  

(10)

For the case with side information, the probability of channel symbol erasure is again the probability that the channel amplitude during the bit transmission time is less than \( \lambda_s \).

\[ P_{ce} = \int_0^\infty p_{ce}(a) p_a(a) \, da. \]  

(11)

In both cases the probability of code symbol erasure is the probability that, among \( m/b \) consecutive channel symbols, at least one symbol is erased. The probability of code symbol erasure for both cases is then

\[ P_e = \begin{cases} 
1 - (1 - p_{ce})^{m/b}, & \text{no side information} \\
1 - (1 - p_{ce}^{SI})^{m/b}, & \text{side information}.
\end{cases} \]  

(12)

The probability of channel symbol error for the case without side information is the expected value of the channel error probability for fixed channel amplitude \( a \).

\[ p_{ce} = \int_0^\infty p_{ce}(a) p_a(a) \, da. \]  

(13)

The probability of code symbol error is then the probability that none of the channel symbols is erased and at least one channel symbol is in error. For the case with side information, the limits of integration are changed to reflect the erasure threshold.

\[ p_{ce}^{SI} = \int_0^\infty p_{ce}^{SI}(a) p_a(a) \, da. \]  

(14)

\[ p_{ce}^{SI} = \int_0^\infty p_{ce}^{SI}(a) p_a(a) \, da. \]  

Fig. 4 compares the code symbol error probabilities for the channel and code symbol interleaved channels. These curves indicate an increase in the probability of symbol error that corresponds to an effective reduction in \( E_b/N_0 \) of 1 to 3 dB when eight-bit code symbols are interleaved bit-by-bit instead of symbol-by-symbol. In the examples in Section IV it will be shown that this reduction has a substantial impact on the decoder error probability for Reed–Solomon codes.

---

Fig. 4. Probability of symbol error curves for eight-bit code symbols.
III. REED–SOLOMON HYBRID-ARQ PROTOCOLS

A type-1 hybrid-ARQ protocol encodes data packets for both error detection and error correction. The decoder uses the error correction capability to correct the most frequently occurring error patterns, while the residual detection capacity is used to detect less frequently occurring patterns. In the event of the latter, or a decoder failure, a retransmission request is sent back to the transmitter via the feedback channel. This form of hybrid-ARQ protocol can be realized using two codes, one for error correction, the other for error detection. It is also possible to perform both the correction and detection functions using a single encoder/decoder pair. The resulting system is, in general, easier to implement for in most cases it is equivalent in complexity and power usage to the error correcting portion of the two codec system. In this paper a single codec Reed–Solomon hybrid-ARQ system is discussed that requires virtually no increase in complexity over the corresponding FEC Reed–Solomon codec.

In an earlier paper [1] a method was demonstrated for modifying FEC Reed–Solomon error control systems for use in type-1 hybrid-ARQ protocols. In this section the earlier method is extended to allow for erasure decoding. The resulting Reed–Solomon Hybrid-ARQ protocol (RS/HARQ) is then shown to provide good reliability performance at the expense of a negligible to moderate reduction in throughput within well defined $E_b/N_0$ operating regions.

One of the most widely used decoding techniques for Reed–Solomon codes is the Berlekamp–Massey algorithm (BMA). The BMA provides bounded distance decoding and is readily extended to provide erasure decoding.

Given a Reed–Solomon code with minimum distance $d_{\text{min}}$, this algorithm can correct all received words containing $e$ symbol errors and $s$ symbol erasures within the constraint $(2e + s) < d_{\text{min}}$. The BMA uses an iterative approach to generate an error location polynomial whose roots indicate the positions of the errors in the received word. An erasure location polynomial is then generated using the erasure locations indicated by the demodulator. Both polynomials are used in conjunction with an error/erasure magnitude polynomial to compute valid symbol values for the erroneous and erased coordinates in the received word. The details of this process are described in a variety of places in the literature (e.g., [3], [9]–[11]). For the purpose of this paper it is only important to note the following. If the received word is within $e$ errors and $s$ erasures of a valid code word and $(2e + s) < d_{\text{min}}$, then the decoder will output that code word. If the selected code word is not the code word that was transmitted, then a decoder error has occurred. If there is no code word within $e$ errors and $s$ erasures, where $(2e + s) < d_{\text{min}}$, then a decoder failure is declared. In the event that a code word is selected by the decoder, the values of $e$ and $s$ can be obtained by examining the degrees of the error and erasure location polynomials respectively.

The underlying principle of the RS/HARQ protocol is that for a given completed decoding operation, the probability that the decoder has made a mistake is proportional to the values of $e$ and $s$. Reliability performance can thus be increased by reducing the number of errors and erasures to be corrected below some threshold and requesting a retransmission of the code word whenever the threshold is exceeded or a decoder failure is declared.

This simple idea is readily reduced to a realizable form. Let $d_e$ be defined as the effective diameter of the RS/HARQ decoder. The effective diameter is the maximum value of the sum $(2e + s)$ for which decoding will be completed. $d_e$ must thus be an integer in the range $[0, d_{\text{min}} - 1]$. Whenever $(2e + s) > d_e$, or any time a decoder failure occurs, a retransmission will be requested. $d_e$ thus defines the balance between error correction and error detection in the RS/HARQ system.

Fig. 6 shows the channel model that will be used in the following performance analysis. The expressions for computing $p_e$ and $p_s$ were derived in the previous section for both the code symbol and channel symbol interleaved Rayleigh fading channels. (In the nonerasure decoding case, $p_s$ is set to zero in the model.) It is assumed that incorrect code symbols are equally probable.

The reliability and throughput performance of the RS/HARQ system can be determined through a series of combinatorial exercises. Consider the case of an $(n, k, d_{\text{min}})$ Reed–Solomon

---

3Since a complete weight enumerator (i.e., one that states how many times each nonzero symbol occurs in each code word) has not yet been found for the RS codes, some type of simplifying assumption is necessary. The worst case approach sets the probability of occurrence for each incorrect symbol equal to $p_s$, the probability of symbol error. This is overly pessimistic, for 1) a fade affecting one channel symbol is likely to corrupt the other channel symbols comprising the code symbol (code symbol interleaved case), and 2) given the large number of low weight nonzero code words, the nonzero code symbol distribution in the most likely error events is approximately uniform.
code used in the RS/HARQ protocol. Since Reed–Solomon codes are linear, one may assume without loss of generality that the all-zero code word has been transmitted. Let \( P_{d_e}^j \) be the probability that a received word is within the decoding sphere of effective diameter \( d_e \) surrounding a code word of weight \( j \). If simple error correction is to be performed without erasure decoding, \( P_{d_e}^j \) takes on the value

\[
P_{d_e}^j = \sum_{w=0}^{\frac{j}{2}} \sum_{v=0}^{w} \binom{n-v}{w} \left( \frac{2^m-1}{2^m} \right)^{w-j} \left( 1 - P_e \right)^{n-j-v} P_e^{n-v-w}.
\]

If erasure decoding is used, \( P_{d_e}^j \) takes on the value

\[
P_{d_e}^j = \sum_{v=0}^{\frac{j}{2}} \sum_{w=0}^{n-v} \sum_{x=0}^{\frac{j-v}{2}} \binom{n-j}{w} \binom{j-x}{y} \left( 1 - P_e \right)^{n-j-v} P_e^{n-v-w}.
\]

Both (16) and (17) are derived in the Appendix. The weight distribution of Reed–Solomon codes is known to be [9]

\[
A_j = \binom{n}{j} (2^m-1) \sum_{i=1}^{j-d_{\min}} (-1)^i \binom{j-1}{i} 2^m(j-i-d_{\min}).
\]

The probability that the decoder with effective diameter \( d_e \) will make a decoder error is thus

\[
P_E = \sum_{j=1}^{n} A_j P_{d_e}^j.
\]

A retransmission request will be generated whenever the received word is not within the decoding sphere surrounding the correct or any one of the incorrect code words. For the nonerasure and erasure decoding cases the following expressions result (see (20) below):

The probability of word error \( P_{WE} \) for packets sent to the data sink is a function of both the decoder error probability \( P_E \) and the probability of retransmission \( P_R \). This is due to the fact that multiple transmissions of the same packet allow the decoder multiple opportunities to make a mistake. It is shown in [2] that the probability of word error among accepted packets is

\[
P_{WE} = \frac{P_E}{1 - P_R}. \quad (21)
\]

The throughput for the RS/HARQ system is a function of the retransmission protocol selected. If a selective-repeat protocol is used, as is assumed in the following examples, then the throughput can be shown to be [2]

\[
\eta = R(1 - P_R). \quad (22)
\]

where \( R \) is the rate of the code in use.

IV. PERFORMANCE EXAMPLES

In the following examples the performance of the RS/HARQ system is examined for a coherent BPSK modem used over a code symbol interleaved slowly fading Rayleigh channel. The performance of the RS/HARQ system over a channel symbol interleaved channel is degraded because of the 1 to 2 dB decrease in the effective \( E_b/N_0 \) (see Fig. 4). It is assumed that side information is available for the declaration of erased channel symbols.

A. The Impact of Erasure Decoding

The first set of examples considers the impact of a variation in the erasure threshold \( \lambda_e \). In Fig. 7–9 the performance of a (16, 12) Reed–Solomon code in the RS/HARQ system is examined. Fig. 7 shows the variation in the probability of word error \( P_{WE} \) as the erasure threshold is increased. It is clear that a significant amount of improvement in reliability performance can be obtained through erasure decoding in the RS/HARQ system at medium to high \( E_b/N_0 \), but very little improvement is obtained at smaller values of \( E_b/N_0 \).

It should be noted that these word error rate curves and those that follow assume that the hybrid-ARQ protocol allows for an unlimited number of retransmission attempts. As a result, the word error rate in all cases approaches unity as the signal to noise ratio is reduced. The performance of a system that employs retry limits can only be understood through comparison of the word error rate curves with the throughput.
Fig. 7. Probability of word error for a (16, 12) Reed–Solomon code in the RS/HARQ system with $d_e = 4$.

Fig. 8. Probability of retransmission for a (16, 12) Reed–Solomon code in the RS/HARQ system with $d_e = 4$.

Fig. 9. Throughput for a (16, 12) Reed–Solomon code in the RS/HARQ system with $d_e = 4$.

Fig. 10. Probability of word error for a (64, 56) Reed–Solomon code in the RS/HARQ system with $d_e = 8$.

Fig. 11. Probability of retransmission for a (64, 56) Reed–Solomon code in the RS/HARQ system with $d_e = 8$.

Fig. 12. Throughput for a (64, 56) Reed–Solomon code in the RS/HARQ system with an effective diameter of eight.

curves that follow. It is then seen that for extremely low signal-to-noise ratios the throughput is essentially zero, so that given a reasonable retry limit, it is highly improbable that erroneous words are accepted by the decoder.

Fig. 8 shows the probability of retransmission $P_R$ for the (16, 12) code as a function of $E_b/N_0$ and the erasure threshold $\lambda_e$. The floor effect is due to the fact that the probability of erasure is independent of the signal to noise ratio on the channel. After a certain point, any further reduction in the probability of symbol error is negated by the probability that the number of erasures alone will be sufficient to exceed the effective diameter of the decoder.

Fig. 9 shows that, all else being equal, the RS/HARQ system throughput for the (16, 12) Reed–Solomon code is only affected by large values of $\lambda_e$. This is apparent in Fig. 8, for though there are floors on the probability of retransmission curves, the floors themselves are at relatively low values for $P_R$. $P_R$ must take on values above 0.1 before any appreciable reduction in throughput can be seen.

Figs. 10–12 show the performance of the (64, 56) Reed–Solomon code in the RS/HARQ system with an effective diameter of eight. Fig. 10 shows the same asymptotic trends seen in Fig. 7, but also shows that higher values of the erasure threshold can actually cause performance to degrade at low signal to noise ratios. The reason for this is seen in Fig. 11: for the cases $\lambda_e = 0.4$ and 0.5 the probability of word error is being significantly increased by the large number of attempts required before the packet is accepted by the receiver (note the denominator in (21)). As $E_b/N_0$ increases, however, a point is reached beyond which the higher erasure thresholds provides better reliability performance than the lower erasure thresholds.

Fig. 12 shows the drastic reduction in throughput caused by the high $P_R$ floors in Fig. 11. It is interesting to note that in the middle range of $E_b/N_0$, the throughput curves for the higher erasure thresholds is temporarily better than that for the lower values. This is because the probability of symbol error is reduced in this erasure generation system by increasing the range of channel amplitudes which will cause erasures. This
effect is not seen elsewhere in the curves because it is masked by other factors.

B. The Impact of the Type-I Hybrid-ARQ Protocol

In this section examples are provided to show the impact of the setting of the effective diameter \( d_e \) in the RS/HARQ protocol. Fig. 13 shows several curves depicting the word error rate for the (64, 56) RS/HARQ system with an erasure threshold \( \lambda_e = 0.1 \) and various values of \( d_e \). The impact of the conversion of the FEC BMA to a hybrid-ARQ protocol is readily apparent. Throughout the operating range of the decoder, each incremental reduction in \( d_e \) results in a reduction of several orders of magnitude in the word error rate. Fig. 14 shows the word error rate for the same values of \( d_e \) when the erasure threshold has been increased to \( \lambda_e = 0.4 \). In this case the reliability performance is even more sensitive to a decrease in the effective diameter of the decoder. Figs. 15 and 16 show the price paid for the improvement in reliability performance; the throughput drops significantly in the middle of the operating range with each successive decrease in \( d_e \). It should be noted that for the case \( \lambda_e = 0.1 \), the setting of \( d_e \) does not affect the asymptotic throughput ceiling established by the erasure threshold (recall Fig. 12). For the case \( \lambda_e = 0.4 \), however, the successive reductions in \( d_e \) cause large reductions in the throughput ceiling. In this case the average number of erasures seen per received packet is approaching the effective diameter of the decoder, causing repeated retransmission requests.

V. CONCLUSION

Expressions for the probabilities of channel symbol error and erasure were derived for slowly fading Rayleigh channels. Two methods for generating channel symbol erasures were considered, the two being differentiated by the existence or nonexistence of forward channel amplitude side information. The resulting probability expressions were then translated into code symbol error and erasure probability expressions for use in evaluating the performance of nonbinary error control codes over code symbol and channel symbol interleaved channels. It was shown that code symbol interleaving offers significantly better symbol error and erasure performance than channel symbol interleaving.

The RS/HARQ error control system for fading channels with feedback was then presented. This system is based on the use of Reed-Solomon codes with erasure decoding in a type-I hybrid-ARQ protocol. The RS/HARQ system requests retransmissions whenever a received word falls outside of the decoding sphere defined by the effective diameter. It was shown that this system provides a substantial improvement in reliability performance at the expense of a reduction in throughput. In conjunction with a variable threshold erasure generation system, the RS/HARQ system provides an extremely powerful and flexible means for controlling errors on a Rayleigh fading channel.
Fig. 16. Throughput for a (64,56) Reed–Solomon code in the RS/HARQ system with $\lambda_s = 0.4$.

**APPENDIX**

**DERIVATION OF (16) AND (17)**

Equation (17) shall be derived first, followed by (16). We wish to determine $P_{d_s}$, the probability that a received word $R$ falls within a decoding sphere of effective diameter $d_s$ surrounding a code word $C_j$ of weight $j$. It shall be assumed that the all-zero code word has been transmitted. It is also assumed that sufficient code symbol interleaving has been employed to render the communication channel memoryless. Let $p_0$ be the probability of correct symbol reception (i.e., a zero symbol is received), $p_e$ the probability of incorrect symbol reception (a zero symbol is received), and $p_s$ the probability of symbol erasure.

Let $\theta$ be the set containing the $(n-j)$ coordinates of $C_j$ that contain zeros. Let $\phi$ be the set containing the $j$ coordinates of $C_j$ that contain nonzero symbols. The desired probability expression can be obtained by allocating $e$ errors and $s$ erasures among the two sets of coordinates in all possible combinations within the constraint $(2e + s) \leq d_s$.

There are five distinct events that must be accounted for in this derivation:

- A $\theta$ or $\phi$ coordinate in $R$ contains a zero symbol. This event occurs with probability $p_0 = (1 - p_s - p_e)$.
- A $\theta$ or $\phi$ coordinate in $R$ contains an erasure. This event occurs with probability $p_s$.
- A $\theta$ coordinate in $R$ contains a nonzero symbol. This event occurs with probability $p_e$.
- A $\phi$ coordinate in $R$ contains the same nonzero symbol as at the same coordinate in $C_j$. This event occurs with probability $p_e/(2^m - 1)$.
- A $\phi$ coordinate in $R$ contains a nonzero symbol that is different from the symbol at the same coordinate in $C_j$. This event occurs with probability $p_e(2^m - 2)/(2^m - 1)$.

The allocation of errors and erasures in the expression is controlled by five counting variables as follows:

- $v$ number of $\theta$ coordinates for which $R$ has a nonzero symbol
- $w$ number of $\theta$ coordinates for which $R$ has an erasure
- $x$ number of $\phi$ coordinates in which $R$ has a nonzero symbol other than the nonzero symbol in $C_j$
- $y$ number of $\phi$ coordinates in which $R$ has an erasure
- $z$ number of $\phi$ coordinates in which $R$ has a zero symbol

$P_{d_s}$ is computed by summing over all possible error/erasure patterns for the all-zero code word such that $(v + w + x + y + z) \leq d_s$. Using the substitution $p_0 = (1 - p_e - p_s)$, the following results.

$$P_{d_s} = \left\{ \begin{array}{l} \sum_{v=0}^{d_s - 2v} \left( \begin{array}{c} n - j \\ v \end{array} \right) p_e^v \sum_{w=0}^{d_s - 2v} \left( \begin{array}{c} n - j - v \\ w \end{array} \right) p_s^w \\ \cdot \left( 1 - p_e - p_s \right)^{n-j-v-w} \cdot \sum_{x=0}^{d_s - 2v - 2w} \left( \begin{array}{c} j \\ x \end{array} \right) \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^{x+y+z} \\ \cdot \left( 1 - p_e - p_s \right)^x \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^y \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^z \end{array} \right\}$$

The case without erasures is obtained in a similar manner. Once again the coordinates in $C_j$ are partitioned into two sets: $\theta$ containing the $(n-j)$ zero coordinates and $\phi$ containing the $j$ nonzero coordinates. The probabilities of the following events are used in the derivation.

- A $\theta$ coordinate in $R$ contains a zero symbol. This event occurs with probability $p_0 = (1 - p_e)$.
- A $\theta$ coordinate in $R$ contains a nonzero symbol. This event occurs with probability $p_e$.
- A $\phi$ coordinate in $R$ contains the same nonzero symbol as at the same coordinate in $C_j$. This event occurs with probability $p_e/(2^m - 1)$.
- A $\phi$ coordinate in $R$ contains a zero symbol or a nonzero symbol that is different from the symbol at the same coordinate in $C_j$. This event occurs with probability $1 - p_e/(2^m - 1)$.

Let the counting variable $v$ denote the number of coordinates that differ between the received word and the code word. The counting variable $w$ shall control the allocation of the $v$ differences between the sets $\phi$ and $\theta$. Using the probabilities of the events listed above, the following expression results:

$$P_{d_s} = \left\{ \begin{array}{l} \sum_{v=0}^{d_s - 2v} \left( \begin{array}{c} n - j \\ v \end{array} \right) p_e^v \left( 1 - p_e \right)^{n-j-v} \\ \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^{x+y+z} \cdot \left( 1 - p_e - p_s \right)^x \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^y \cdot \left( \frac{2^m - 2}{2^m - 1} \right)^z \end{array} \right\}$$

The case without erasures is obtained in a similar manner. Once again the coordinates in $C_j$ are partitioned into two sets: $\theta$ containing the $(n-j)$ zero coordinates and $\phi$ containing the $j$ nonzero coordinates. The probabilities of the following events are used in the derivation.

- A $\theta$ coordinate in $R$ contains a zero symbol. This event occurs with probability $p_0 = (1 - p_e)$.
- A $\theta$ coordinate in $R$ contains a nonzero symbol. This event occurs with probability $p_e$.
- A $\phi$ coordinate in $R$ contains the same nonzero symbol as at the same coordinate in $C_j$. This event occurs with probability $p_e/(2^m - 1)$.
- A $\phi$ coordinate in $R$ contains a zero symbol or a nonzero symbol that is different from the symbol at the same coordinate in $C_j$. This event occurs with probability $1 - p_e/(2^m - 1)$.
REFERENCES


Stephen B. Wicker (S'83–M'83) was born in Hazelhurst, MS, on September 25, 1960. He received the B.S.E.E. degree with high honors from the University of Virginia, Charlottesville, in 1982, the M.S.E.E. degree from Purdue University, West Lafayette, IN, in 1983, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, in 1987.

From 1983 through 1987 he was a subsystem and system engineer with the Space and Communications Group of the Hughes Aircraft Company, El Segundo, CA. In September 1987 he joined the faculty of the School of Electrical Engineering at the Georgia Institute of Technology, where he currently holds the title of Associate Professor. From June 1991 until March 1992 he served as the Academic Coordinator for Georgia Tech-Lorraine in Metz, France. His current research interests center on the development of algorithms for error control, data compression, and data security for digital communication systems. He is the inventor or co-inventor of several adaptive error control algorithms for nonstationary communication channels, many of which are now seeing commercial application. He has served as a consultant in these areas for several telecommunications companies in the United States, Canada, and France. He is the author of over 45 technical publications.

Dr. Wicker is a member of the IEEE Communications, Information Theory, and Vehicular Technology Societies. He is also a member of Eta Kappa Nu, Tau Beta Pi, Sigma Xi, and Omicron Delta Kappa. He was named a Visiting Fellow of the British Columbia Advanced Systems Institute in 1992.
A Sequential Scheme for Adaptive Error Control Over Slowly Varying Channels*

Michael D. Rice and Stephen B. Wicker
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332
March 1, 1991

abstract

Improved performance of error control techniques over slowly varying channels can be realized by a system which adapts itself to changes in the channel conditions. The frequency of retransmission requests in ARQ and type-I hybrid-ARQ error control strategies provides a natural source of channel state information. The proposed scheme incorporates the retransmission requests into the scoring of a statistical sequential inspection scheme proposed by E. Page in 1954. Each transmitted packet is scored based on the outcome of the decoding process (i.e., whether it is accepted or rejected). When the cumulative score crosses a decision boundary, the coding strategy is altered and the sequential inspection scheme is restarted using the same scoring routine with different weighting and boundary constants. In this way the channel encoder/decoder is able to alter its strategy and adapt itself to changes in the channel. The analysis of this system is based on the average run length of the tests in the various channel states. These values indicate the delay the system experiences in reacting to a change in the channel and the tendency the test has to terminate prematurely. These factors are a function of the test constants which are determined algorithmically by maximizing the system throughput using the system reliability as a

*This work was supported by the National Science Foundation under grant number NCR-9009877
constraint. Results show that the performance of this system is close to that of the ideal system with perfect channel state information.

1 Introduction

ARQ and hybrid-ARQ error control strategies are useful in combating transmission errors induced by noisy channels when high system reliability is required. The performance of ARQ and hybrid-ARQ systems is measured in terms of reliability and throughput [1,2]. The reliability, \( P(E) \), is the probability of accepting a packet which contains undetected errors. The throughput \( T \) is a measure of the efficiency of the system and is defined as the expected number of information symbols accepted per transmitted symbol. In general, a system which employs retransmissions as part of its error control strategy exhibits a trade-off between the reliability and the throughput. For a given rate \( k/n \) code at a given channel bit error rate, a reduction in \( P(E) \) is accompanied by a reduction in \( T \), and vice versa.

![Figure 1: Throughput Comparisons of Various Error Control Schemes.](image)

The error processes causing transmission errors are rarely constant over time [3]. This results in channels which are at times well behaved and at other times noisy. Consequently,
the performance of a system employing retransmissions in an error control strategy varies greatly resulting in a reduction in the efficiency of the system. To illustrate this point, consider the performance of length 127 BCH codes over a binary symmetric channel with a selective-repeat retransmission protocol illustrated in Figure 1. The ARQ coding strategy uses the (127,106) BCH code and the FEC strategy is provided by the (127,78) BCH code. The hybrid-ARQ error control utilizes the (127,99) BCH code with a decoder allowed to correct up to 2 errors. In each case the error control strategy guarantees a probability of undetected error $\leq 10^{-6}$ over all ranges of channel bit error rates. However the throughputs are seen to vary dramatically as the channel bit error rate varies. This suggests the use of different error control strategies for different channel conditions. For example, when the channel is clear, it is seen that the ARQ strategy offers the best throughput. When the channel is noisy, FEC offers the highest throughput. Hybrid-ARQ strategies are best suited for use when the channel is somewhere between the two extremes. For a channel with varying error processes, the use of a single coding strategy will not yield the optimal throughput for a given level of reliability.

To compensate for slow fluctuations in channel error processes, adaptive error control techniques based on ARQ and type-I hybrid-ARQ protocols have been proposed [4,5,6,7,8,9]. The channel is modeled as an $M$-state Markov chain ($M$ is finite) and for each channel state $m$ an error control strategy $C_m$ is selected. The general structure of these systems is illustrated in Figure 2. Data to be transmitted originates at the data source and is encoded using one of the $M$ code rates by the forward channel encoder block. The transmission errors occur in the forward channel, which is composed of the transmitter, the physical channel, and the receiver. The received packet is decoded by the forward channel decoder based on an ARQ or hybrid-ARQ rule.

In ARQ and type-I hybrid-ARQ strategies, the decoding results are completely described by two events:

1. the packet is accepted
2. the packet is rejected.

Fluctuations in the channel conditions are detected by monitoring the rate at which retransmission requests are generated. This function is performed by the block labeled

---

$^{1}$The (127,99) BCH code has a minimum distance of 9 thus enabling it to correct 4 errors. Since the code is allowed to correct only 2 errors, it can detect $v$ errors where $2 < v < 7$ thereby defining a hybrid-ARQ protocol.

---
"channel state monitor." When a change in the channel state has been determined, the encoder/decoder pair selects a different error control strategy to compensate for the change.

The problem of detecting a change in the channel state may be viewed in a different manner by defining a random variable $X$ mapping the space of possible decoder outcomes to the set $\{0, 1\}$ as follows:

$$X = \begin{cases} 
0 & \text{if the packet is accepted} \\
1 & \text{if the packet is rejected} 
\end{cases}$$

Thus $X$ is described by the probability

$$Pr\{X = x; R\} = R^x(1 - R)^{1-x} \; \text{for} \; x \in \{0, 1\}$$

where the parameter $R$ is the probability of retransmission (i.e. the probability that the packet is rejected). The problem then reduces to that of finding a statistical test that will detect a change in the parameter $R$ of the distribution of $X$ by considering samples $x_1, x_2, \ldots$ of $X$. In the terminology of hypothesis testing, the null hypothesis $H_0$, defined by $R = R_0$, represents the belief that the parameter $R$ has remained unchanged and has a satisfactory value and the alternative hypothesis $H_1$ represents a change in $R$ to some value $R_1 \neq R_0$.

Most of the proposed systems use a "fixed sample size" test to perform this task. Typically, the channel is monitored by counting the number of retransmissions during an observation interval $N$ and comparing that number with a set of thresholds to determine
the channel state. For example if a coding strategy denoted \( C_2 \) is being used, the number of retransmissions \( \eta \) is counted during the observation interval. At the end of the observation interval, \( \eta \) is compared to a set of thresholds. If \( \eta < \tau_{2,1} \) then \( R \) has assumed a lesser value and a different coding strategy \( C_1 \) will be used. If \( \tau_{2,1} \leq \eta < \tau_{2,3} \) then it is assumed that \( R \) has remained unchanged and \( C_2 \) is used during the next observation interval. Likewise, if \( \eta \geq \tau_{2,3} \) then \( C_3 \) will be used. If \( N \) is large, then the delay in reacting to a change in the channel could be significant resulting in a degradation in performance. If, on the other hand, \( N \) is too small, the channel state monitor will not have enough information to make a reliable decision. Frequent erroneous decisions of this nature also degrade performance.

This paper presents a sequential method of performing the channel state monitoring function. A sequential test differs from a fixed sample test in that the sample size required to make a decision about the truth of a hypothesis is not fixed in advance but allowed to vary depending on the observations. The number of samples required to make a decision is often less than the number required by a fixed sample size counterpart [10].

## 2 Sequential Tests

A Wald sequential test is used to discriminate between two simple hypotheses \( H_0 \) and \( H_1 \) by calculating a function \( S_n(X)^\dagger \) of \( x_1, x_2, \ldots \) after each sample \( n \) and performing the following test:

\[
\begin{align*}
\text{reject } H_0 & \quad \text{if } S_n(X) \geq A \\
\text{accept } H_0 & \quad \text{if } S_n(X) \leq B \\
\text{take another sample} & \quad \text{if } B < S_n(X) < A.
\end{align*}
\]

where \(-\infty < B < A < \infty\) are constants which form the "boundaries" of the test. Note that the tests decides to accept one of the hypothesis only when one of the first two of the conditions in (1) is satisfied. The performance of sequential tests is measured by two quantities called the operating curve and the average sample number (ASN) [10,11]. The operating curve is the probability that the test ends in acceptance. It is a function of \( Pr\{X = x; R\} \) which is in turn a function of the parameter \( R \). Hence the operating curve of a Wald test is denoted \( P(R) \) to emphasize this functional dependence. The number of observations or samples required by the test to reach a decision is a random variable which depends on \( R \) through \( Pr\{X = x; R\} \). The expected value of this random variable is the average sample number and is denoted \( N(R) \).

\[\dagger\]\( S_n(X) \) is usually taken to be the log-likelihood ratio, but this is not a requirement.
In 1954 E. S. Page proposed a continuous inspection scheme designed to detect a change in a parameter describing a stochastic process [12]. It has been shown that Page’s "stopping rule" is optimum in a well defined sense [13]. If the null hypothesis is accepted, then no action is required and the process is allowed to continue. If the null hypothesis is rejected, the process has changed and action is required to be taken. The key issue here is to determine a change in the parameter $R$ quickly without too many "false alarms." In this scheme, the outcome of each sample is scored and a stopping rule is defined. The process is allowed to continue as long as $H_0$ is believed to be true whereas action is taken when $H_0$ is rejected.

The scoring rule is implemented by defining a new random variable $z_i$ by scoring the value assumed by each of the $x_i$ as follows [12]:

$$
z_i = \begin{cases} 
-1 & \text{if } x_i = 0 \\
+b & \text{if } x_i = 1 
\end{cases} \quad (2)
$$

for some positive integer $b$ and summing the $z_i$ to form $S_n$:

$$
S_n = \sum_{i=1}^{n} z_i.
$$

The stopping rule for the case $R_1 > R_0$ is defined as follows:

$$
\text{Reject } H_0 \text{ if } S_n - \min_{0 \leq i \leq n} S_i \geq h 
$$

after the $n^{th}$ observation where $h$ is a positive integer. As Page illustrated, this scheme is equivalent to a series of Wald tests where $S_n(X) = S_n$ with acceptance boundary 0, rejection boundary $h$, and initial score 0. The continuous inspection scheme is performed by restarting the Wald test each time the test ends in acceptance and rejecting the null hypothesis when the test ends on or above the rejection boundary.

The stopping rule for the case $R_1 < R_0$ is similarly defined:

$$
\text{Reject } H_0 \text{ if } \max_{0 \leq i \leq n} S_i - S_n \geq k 
$$

after the $n^{th}$ observation where $k$ is a positive integer constant. This scheme is equivalent to a series of Wald tests with acceptance boundary 0, rejection boundary $-k$, and initial score 0. The continuous inspection scheme is performed by restarting the Wald test each time the test ends in acceptance and terminates when the Wald test ends in rejection.
The above tests are examples of *one sided tests* designed to detect a change in \( R \) in one direction only. To detect changes in \( R \) in both directions, a the simultaneous application of the above two schemes is used. The stopping rule is

\[
\text{Reject } H_0 \text{ if either } S_n - \min_{0 \leq i \leq n} S_i \geq h \text{ or } \max_{0 \leq i \leq n} S_i - S_n \geq k
\]

(5)
after the \( n \)th observation where, as before, \( h \) and \( k \) are positive integer constants. This scheme is equivalent to the simultaneous application of a series of two Wald sequential tests with \( S_n(X) = S_n \). The first has boundaries \((0, h)\) and starts on the lower boundary. The second has boundaries \((-k, 0)\) and starts on the upper boundary.

The performance of each inspection scheme \( Q \) is described by its the average run length (ARL) \( L(R) \) which is the expected value of the number of samples observed before the null hypothesis is rejected. \( L(R) \) may be expressed in terms of the operating curve \( P(R) \) and the ASN, \( N(R) \), of its constituent Wald tests. Let \( N_0(R) \) represent the ASN conditional on the constituent Wald test ending on the acceptance boundary and let \( N_1(R) \) be the ASN conditional the test ending on the rejection boundary. Clearly

\[
N(R) = P(R)N_0(R) + [1 - P(R)]N_1(R).
\]

Denote by \( \rho \) the number of acceptance tests before action is taken. Then

\[
Pr\{\rho = r\} = P^r(R)[1 - P(R)],
\]

and the expected value of \( \rho \) is given by

\[
E\{\rho\} = \sum_{r=1}^{\infty} rP^r(R)[1 - P(R)] = [1 - P(R)]\sum_{r=1}^{\infty} rP^r(R) = \frac{P(R)}{1 - P(R)}.
\]

(6)

The ARL is thus given by

\[
L(R) = \frac{P(R)}{1 - P(R)}N_0(R) + N_1(R) = \frac{P(R)N_0(R) + [1 - P(R)]N_1(R)}{1 - P(R)} = \frac{N(R)}{1 - P(R)}.
\]

(7)
Equation (7) shows that the ARL of the sequential inspection scheme \( Q \) is a function of both the ASN and the operating curve of the Wald tests which constitute the scheme \( Q \). Expressions for the ARL's of the three types of inspection schemes will now be derived.

Burman [14], Walker [15], and Girshick [16] have analyzed the Wald test with boundaries \((0, H)\) and the following scoring

\[
z_i = \begin{cases} 
+1 & \text{if } x_i = 0 \\
-b & \text{if } x_i = 1 
\end{cases}
\]

where \( H \) is the acceptance boundary. It is seen that this scoring is identical to but opposite in sign to that of the Wald tests considered previously. Let \( P(z; R) \) denote the probability that the test ends in acceptance when the running sum \( S_n \) is \( z \). The resulting difference equation is solved for \( P(z; R) \) and is given by

\[
P(z; R) = \frac{F(z, b; R)}{F(H, b; R)}
\]

where

\[
F(z, b; R) = (1 - R)^{1-z} \sum_{j=0}^{[\frac{z+1}{b}]} (-1)^j \binom{z - jb - 1}{j} (R(1 - R)b)^j.
\]

The ASN when \( S_n = z \) is given by

\[
A(z, b, H; R) = \frac{1}{R} \left\{ \frac{F(z, b; R)}{F(H, b; R)} \left[ \sum_{i=1}^{K_1} F(H - ib, b; R) - (K_1 + 1) \right] \\
- \left[ \sum_{i=1}^{K_2} F(z - ib, b; R) - (K_2 + 1) \right] \right\}
\]

where

\[
K_1 = \left\lfloor \frac{H - 1}{b} \right\rfloor \quad \text{and} \quad K_2 = \left\lfloor \frac{z - 1}{b} \right\rfloor.
\]

Applying (9) and (11) to the Wald tests of the increasing inspection scheme (3) when \( S_n = z \) yields the following expressions:

\[
P(z; R) = \frac{F(h - z, b; R)}{F(h, b; R)}
\]
\[
N(z; R) = A(h - z, b, h; R)
\]

\[
= \frac{1}{R} \frac{F(h - z, b; R)}{F(h, b; R)} \left[ \sum_{i=1}^{K_1} F(h - ib, b; R) - (K_1 + 1) \right] \\
- \left[ \sum_{i=1}^{K_2} F(h - z - ib, b; R) - (K_2 + 1) \right]
\]
where
\[ K_1 = \left[ \frac{h - 1}{b} \right] \text{ and } K_2 = \left[ \frac{h - z - 1}{b} \right]. \]

It is necessary to modify these equations for they do not hold on the acceptance boundary. This is accomplished by taking the expected value conditional on the first observation [12):

\[
P(0; R) = (1 - R)P(-1; R) + RP(b; R)
\]
\[
= (1 - R) + RP(b; R)
\]
\[
= (1 - R) + R \frac{F(h - b, b; R)}{F(h, b; R)}
\]
\[
N(0; R) = (1 - R)N(-1; R) + RN(b; R)
\]
\[
= 1 + RN(b; R)
\]
\[
= 1 + \frac{1}{R} \sum_{i=1}^{K_1} F(h - ib, b; R) - (K_1 + 1)
\]
\[
- \left[ \sum_{i=1}^{K_2} F(h - ib, b; R) - (K_2 + 1) \right]
\]

with
\[ K_1 = \left[ \frac{h - 1}{b} \right] \text{ and } K_2 = \left[ \frac{h - b - 1}{b} \right]. \]

The ARL of the increasing test is thus given by
\[
L_+(h, b; R) = \frac{N(0; R)}{1 - P(0; R)}
\]
\[
= 1 + RN(b; R)
\]
\[
= 1 - [(1 - R) + RP(b; R)]
\]
\[
= 1 + RN(b; R)
\]
\[
= R[1 - P(b; R)]
\]
\[
= \frac{1}{R} \left[ 1 - \frac{F(h-b,b;R)}{F(h,b;R)} \right]
\]
\[
\cdot \left\{ 1 + \frac{F(h-b,b;R)}{F(h,b;R)} \left[ \sum_{i=1}^{K_1} F(h - ib, b; R) - (K_1 + 1) \right] \right. \]
\[
- \left[ \sum_{i=1}^{K_2} F(h - ib, b; R) - (K_2 + 1) \right] \}
\]

where again
\[ K_1 = \left[ \frac{h - 1}{b} \right] \text{ and } K_2 = \left[ \frac{h - b - 1}{b} \right]. \]
Applying (9) and (11) to the Wald tests of the decreasing inspection scheme (4) for 
\(S_n = -z\), the performance is given by

\[
P(-z; R) = 1 - \frac{F(z, b; R)}{F(k, b; R)}
\]

\[
N(-z; R) = A(-z, b, k; R)
\]

\[
= \frac{1}{R} \frac{F(z, b; R)}{F(k, b; R)} \left[ \sum_{i=1}^{K_1} F(k - ib, b; R) - (K_1 + 1) \right] 
- \left[ \sum_{i=1}^{K_2} F(z - ib, b; R) - (K_2 + 1) \right]
\]

where in this case

\[
K_1 = \left\lfloor \frac{k - 1}{b} \right\rfloor \quad \text{and} \quad K_2 = \left\lfloor \frac{z - 1}{b} \right\rfloor.
\]

It will again be necessary to modify these equations since they do not apply on the starting boundary. As before, taking the expected value conditioned on the outcome of the first observation

\[
P(0; R) = (1 - R)P(-1; R) + RP(b; R)
\]

\[
= (1 - R)P(-1; R) + R
\]

\[
= (1 - R) \left[ 1 - \frac{F(1, b; R)}{F(k, b; R)} \right] + R
\]

\[
= 1 - (1 - R) \frac{F(1, b; R)}{F(k, b; R)}
\]

(15)

\[
N(0; R) = (1 - R)N(-1; R) + RN(b; R)
\]

\[
= (1 - R)N(-1; R) + 1
\]

\[
= \frac{1 - R}{R} \left\{ \frac{F(1, b; R)}{F(k, b; R)} \left[ \sum_{i=1}^{K} F(k - ib, b; R) - (K + 1) \right] + 1 \right\} + 1
\]

(16)

where

\[
K = \left\lfloor \frac{k - 1}{b} \right\rfloor.
\]

The ARL for the decreasing test is therefore

\[
L_-(k, b; R) = \frac{N(0; R)}{1 - P(0; R)}
\]

\[
= \frac{(1 - R)N(-1; R) + RN(b; R)}{1 - [(1 - R)P(-1; R) + R]}
\]
\[
= \frac{1-R}{R} \left\{ \frac{F^{(1,b,R)}_{(k,b,R)}}{F^{(1,b,R)}_{(k,b,R)}} \left[ \sum_{i=1}^{K} F(k-i\,b,b,R) - (K + 1) \right] + 1 \right\} + 1
\]

where again

\[ K = \left\lfloor \frac{k-1}{b} \right\rfloor. \]

Consider the general case of a two-sided Wald test illustrated in Figure 3. In this illustration, \( H_+ \) is the hypothesis that \( R = R_+ \) for some \( R_+ > R \) and \( H_- \) is the hypothesis that \( R = R_- \) for some \( R_- < R \). \( H_0 \) is the hypothesis that \( R \) has not changed from \( R_0 \). When the test ends in the region denoted \( H_0 \), the test is restarted. This region represents the acceptance region for both tests. The following probabilities are defined to analyze the test:

\[ P(H_+) = \text{the probability that the test ends in region } H_+ \]

\[ P(H_0) = \text{the probability that the test ends in region } H_0 \]

\[ P(H_-) = \text{the probability that the test ends in region } H_- \]

Clearly \( P(H_+) + P(H_0) + P(H_-) = 1 \) since the three hypotheses completely describe all states of nature. \( P(H_+) \) and \( P(H_-) \) may be expressed in terms of the operating curves of the increasing and the decreasing tests denoted \( P_+(0;R) \) and \( P_-(0;R) \), respectively. From the definition of the operating curve it is seen that

\[ P(H_+) = 1 - P_+(0;R) \]

\[ P(H_-) = 1 - P_-(0;R). \]
Thus the operating curve for the two-sided test is

\[ P(H_0) = 1 - P(H_+) - P(H_-) \]

\[ = P_+(0; R) - (1 - P_-(0; R)) \]

\[ = (1 - R) + R \frac{F(h - b, b; R)}{F(h, b; R)} - (1 - R) \frac{F(1, b; R)}{F(k, b; R)}. \]  

(18)

In general the ARL of a two-sided test is difficult to obtain. In this case however the analysis is simplified by the fact that the tests have a common starting point (namely \( S_0 = 0 \)) and are independent after the first observation. Page showed that in this case the ARL of the two-sided test may be expressed in terms of the ARL’s of the increasing and decreasing tests which compose the two-sided test [12]. Let \( N_+(0; R) \) and \( N_-(0; R) \) be the ARL’s of the increasing and decreasing tests, respectively, and denote by \( N(O; R) \) the ARL if the two-sided test. Then

\[ N(0; R) = \max \{N_+(0; R), N_-(0; R)\}. \]

Since one of the two tests terminate after the first observation,

\[ \min \{N_+(0; R), N_-(0; R)\} = 1. \]

So

\[ N(0; R) = \max \{N_+(0; R), N_-(0; R)\} + \min \{N_+(0; R), N_-(0; R)\} - 1 \]

\[ = N_+(0; R) + N_-(0; R) - 1 \]  

(19)

and the ARL is given by

\[ L(h, k, b; R) = \frac{N(0; R)}{1 - P(0; R)} \]

\[ = \frac{N_+(0; R) + N_-(0; R) - 1}{1 - \left[ (1 - R) + R \frac{F(h - b, b; R)}{F(h, b; R)} - (1 - R) \frac{F(1, b; R)}{F(k, b; R)} \right]} \]

\[ = \frac{N_+(0; R) + N_-(0; R) - 1}{R \left[ 1 - \frac{F(h - b, b; R)}{F(h, b; R)} \right] + (1 - R) \frac{F(1, b; R)}{F(k, b; R)}} \]  

(20)

where

\[ N_+(0; R) = 1 + \frac{F(h - b, b; R)}{F(h, b; R)} \sum_{i=1}^{K_1} F(h - ib, b; R) - (K_1 + 1) \]

\[ - \sum_{i=1}^{K_2} F(h - b - ib, b; R) - (K_2 + 1) \]

12
and
\[ N_\infty(0; R) = \frac{1 - R}{R} \left\{ \frac{F(1, b; R)}{F(k, b; R)} \left[ \sum_{i=1}^{K} F(k - ib, b; R) - (K + 1) \right] + 1 \right\} + 1. \]

3 Channel Model

Channel models are often described by a set of parameters which remain constant with respect to time. Such models are said to represent "stationary" channels. For example, the traditional binary symmetric channel is a memoryless channel completely described by a single parameter \( p \), the probability of bit error. In general, the parameter necessary for the evaluation of the performance of an error control coding scheme (ARQ or FEC) is \( P(m, n) \), the probability of exactly \( m \) errors occurring in a block of \( n \) symbols. Much work has been done to derive descriptive and generative models of the error producing process [3].

The descriptive parameters of real channels vary over time. If the channel varies slowly in relation to the data signaling rate, then the channel can be modeled as "stationary" over a certain short period of time. A nonstationary channel can thus be modeled as a finite \( M \)-state Markov chain where each state is a stationary channel model [17]. For example, on an AWGN channel with varying signal to noise ratio and hard decision decoding, each state can be modeled as a discrete memoryless symmetric channel with a particular probability of error.

Here it is assumed that the packet lengths are chosen so that the processes defining the error bursts will be constant during the transmission of a single packet but independent from packet to packet. The channel is modeled as having \( M \) states \( 1, 2, \ldots, M \) where channel state \( j \) is defined by the \( P_j(m, n) \) which is the probability of \( m \) errors in a block of \( n \) symbols when the channel is in state \( j \). The channel states are ordered by \( \mu_1 < \mu_2 < \cdots < \mu_M \), where
\[ \mu_j = \sum_{m=0}^{n} mP_j(m, n) \]

Thus the channel state labeled "1" is the best channel and the state labeled "M" is the worst or noisiest. The \( M \)-state Markov chain channel model shown in Figure 4 is represented by the state transition matrix \( P_C = \{p_{ij}\} \), where the \( \{p_{ij}\} \) represent the conditional one step probabilities of going from channel state \( i \) to channel state \( j \) on a packet by packet basis. Note that \( p_{ij} = 0 \) when \( |i - j| > 1 \) for \( i, j = 1, 2, \ldots, M \) so that the channel state transitions are only allowed to adjacent states. It is assumed that the parameters describing the channel vary slowly with time so the transitions from a channel state back to itself should be high (i.e., \( p_{ii} \approx 1 \)).
The slowly varying characteristic of these channels is the result of a slow progression of the conditions which cause the channel noise. For example, the noise on a satellite channel is due primarily to atmospheric conditions such as rain, cloud cover, and humidity. These conditions change slowly in comparison to the signaling rates used in satellite systems. Further, these changes will follow a normal progression of events which correspond to movement from a particular channel state to the adjacent channel state. On switched telephone lines used for data transmission, the noise level is a function of the number of users on the system. The number of users varies slowly in comparison to the signaling rate and is unlikely to jump from a low number to an extremely high number instantaneously. Thus changes in the channel state will normally involve movement to adjacent channel states.

Since the channel is a finite ergodic Markov chain, it has a long run distribution given by \( \pi = (\pi_1, \pi_2, \ldots, \pi_M) \) where \( \pi_i, i = 1, 2, \ldots, M \), may be interpreted as the portion of time the channel is in state \( S_i \) after a "long time". Another parameter of the channel which will be useful in analyzing the performance of the system is the dwell time in channel state \( S_i \) denoted \( D_i \). The dwell time is the expected value of the number of packets received when the channel stays in state \( S_i \). This is equivalent to the number of "steps" the Markov chain stays in state \( S_i \) on average. The dwell time is given by

\[
D_i = \sum_{j=1}^{M} \pi_i P_{ij}
\]
\[ D_i = \sum_{j=1}^{\infty} j Pr\{\text{channel is in state } i \text{ } j \text{ steps} \mid \text{channel in } i \} \]
\[ = \sum_{j=1}^{\infty} j p_{i,j}^{j-1} (1 - p_{i,i}) \]
\[ = (1 - p_{i,i}) \frac{1}{(1 - p_{i,i})^2} \]
\[ = \frac{1}{1 - p_{i,i}}. \tag{21} \]

4 Application of Sequential Tests to Adaptive Error Control

The application of the sequential procedures presented above to the adaptive error control problem is straightforward. It is simply a matter of identifying the action to be taken with the appropriate coding strategy. Recall that the system is equipped with \( M \) error control strategies denoted \( C_1, C_2, \ldots, C_M \). Associated with each \( C_m \) is a sequential inspection scheme denoted \( Q_m \). For \( 1 < m < M \) \( Q_m \) is a two sided continuous inspection scheme and is implemented as shown in Figure 5. This test is used to detect both an improvement and a degradation in the channel. This detection is performed when the test ends in rejection.
The action taken is to use $C_{m+1}$ to encode the next packets and start test $Q_{m+1}$ when the increasing test ends in rejection or to use $C_{m-1}$ to encode the next packets and start test $Q_{m-1}$ when the decreasing test ends in rejection.

Since channel state 1 is the "best" channel state, only a degradation in the channel error rate is possible. Thus an increase in the parameter $R$ is the only possibility in this case. Accordingly, test $Q_1$ is an increasing inspection scheme shown in Figure 6. Only a decrease in the parameter $R$ is possible when the channel is in state $M$. So $Q_M$ is the decreasing inspection scheme shown in Figure 7.

Figure 6: Test $Q_1$ Used With Error Control Strategy $C_1$

![Diagram of test $Q_1$]

Figure 7: Test $Q_M$ Used With Error Control Strategy $C_M$

![Diagram of test $Q_M$]
5 Performance

The performance of the adaptive error control system over a slowly varying $M$ state channel may be evaluated by counting the number of packets transmitted in the possible operating states. This number may be approximated using the ARL’s of the inspection schemes in the corresponding operating states. The following notation is used throughout the analysis:

\[
\begin{align*}
N_{i,j} &= \text{the number of packets transmitted in channel state } ii \\
T_{i,j} &= \text{the throughput in channel state } i \text{ while error control strategy } C_j \text{ is used.} \\
P_{i,j}(E) &= \text{the reliability in channel state } i \text{ while error control strategy } C_j \text{ is used.}
\end{align*}
\]

First, consider a system equipped with an ideal channel state monitor. This monitor has perfect and instantaneous information about the channel. Such a system reacts immediately to any change in the channel conditions and always uses the error control strategy best suited for the channel state. The performance of the ideal system is

\[
\begin{align*}
T_{\text{ideal}} &= \sum_{i=1}^{M} \pi_i T_{i,i} \quad (22) \\
P_{\text{ideal}}(E) &= \sum_{i=1}^{M} \pi_i P_{i,i}(E). \quad (23)
\end{align*}
\]

For the adaptive system, denote by $T_i$ the system throughput when the channel is in state $i$. An approximate expression for $T_i$ may be computed by using the average number of packets received in each of the three possible operating states\(^2\) The average number of packets received while using test $Q_i$ is

\[
N_{i,i} = D_i - L_i - F_i
\]

where $L_i$ is the average number of packets received after a channel state transition to $i$ has occurred before the system detects that change. $F_i$ is the average number of packets received during “false alarms” (i.e. when the channel state remains unchanged but the system thinks it has changed). Taking the expected value of the delay experienced by the system in

\(^2\)The three possible operating states while the channel is in state $i$ are defined as follows: channel state $i$ using test $Q_{i-1}$, channel state $i$ using test $Q_i$, and channel state $S_i$ using test $Q_{i+1}$.
recognizing the change in the channel state, $L_i$ for $1 < i < M$ is calculated as follows:

$$L_i = Pr \{ \text{channel in state } i - 1 \mid \text{channel not in state } i \} \cdot (\text{ARL of } Q_{i-1} \text{ in channel state } i) + Pr \{ \text{channel in state } i + 1 \mid \text{channel not in state } i \} \cdot (\text{ARL of } Q_{i+1} \text{ in state } i)$$

$$= \frac{Pr \{ \text{channel in state } i - 1 \text{ AND channel not in state } i \}}{Pr \{ \text{channel not in state } i \}} \cdot L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{Pr \{ \text{channel in state } i + 1 \text{ AND channel not in state } i \}}{Pr \{ \text{channel not in state } i \}} \cdot L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})$$

$$= \frac{Pr \{ \text{channel in state } i - 1 \}}{Pr \{ \text{channel not in state } i \}} L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{Pr \{ \text{channel in state } i + 1 \}}{Pr \{ \text{channel not in state } i \}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})$$

$$= \frac{\pi_{i-1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1}).$$ (24)

For $1 < i < M$, the average value for the number of false alarm packets is

$$F_i = \text{(average number of false alarms)} \cdot \text{(average number of packets transmitted per false alarm)}$$

$$\approx \left( \frac{D_i}{L(h_i, k_i, b_i; R_{i,i}) + \frac{\pi_{i-1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})} \cdot [Pr \{ Q_i \text{ terminates in } H_- \mid Q_i \text{ terminates} \} \cdot (\text{ARL of } Q_{i-1} \text{ in channel state } i)$$

$$+ Pr \{ Q_i \text{ terminates in } H_+ \mid Q_i \text{ terminates} \} \cdot (\text{ARL of } Q_{i+1} \text{ in channel state } i)$$

$$= \left( \frac{D_i}{L(h_i, k_i, b_i; R_{i,i}) + \frac{\pi_{i-1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})} \right)$$

$$= \left( \frac{D_i}{L(h_i, k_i, b_i; R_{i,i}) + \frac{\pi_1}{\pi_1 + \pi_3} L(h_i - 1, k_i - 1, b_i - 1; R_{i,i-1}) + \frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})} \right)$$
In both (24) and (25), $L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1})$ is replaced by $L_{+}(h_{1}, k_{1}; R_{2,1})$ when $i = 2$ and $L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})$ is replaced by $L_{-}(k_{M}, b_{M}; R_{M,M-1})$ when $i = M - 1$.

The expressions (24) and (25) can be used to compute the values of $N_{i,i-1}$ and $N_{i,i+1}$:

$$N_{i,i-1} = \left(\frac{\pi_{i-1}}{\pi_{i-1} + \pi_{i+1}} + \Delta_{i} \cdot \frac{P_{i}(H_{-})}{P_{i}(H_{-}) + P_{i}(H_{+})}\right) L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) \quad (26)$$

$$N_{i,i+1} = \left(\frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} + \Delta_{i} \cdot \frac{P_{i}(H_{+})}{P_{i}(H_{-}) + P_{i}(H_{+})}\right) L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1}) \quad (27)$$

where $\Delta_{i} = \frac{D_{i}}{L(h_{i}, k_{i}, b_{i}; R_{i,i}) + \frac{\pi_{i-1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1}) + \frac{\pi_{i+1}}{\pi_{i-1} + \pi_{i+1}} L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})}$

where again $L(h_{i-1}, k_{i-1}, b_{i-1}; R_{i,i-1})$ is replaced by $L_{+}(h_{1}, k_{1}; R_{2,1})$ when $i = 2$ and, when $i = M - 1$, $L(h_{i+1}, k_{i+1}, b_{i+1}; R_{i,i+1})$ is replaced by $L_{-}(k_{M}, b_{M}; R_{M,M-1})$.

$T_{i}$ may now be computed for $1 < i < M$:

$$T_{i} = \frac{N_{i,i-1}T_{i,i-1} + N_{i,i}T_{i,i} + N_{i,i+1}T_{i,i+1}}{D_{i}} \quad (28)$$

When $i = 1$

$$T_{1} = \frac{D_{1} - L_{1} - F_{1}}{D_{1}} T_{1,1} + \frac{L_{1} + F_{1}}{D_{1}} T_{1,2} \quad (29)$$

where

$$L_{1} \approx L(h_{2}, k_{2}, b_{2}; R_{1,2}) \quad (30)$$

$$F_{1} \approx \left(\frac{D_{1}}{L_{+}(h_{1}, b_{1}; R_{1,1}) + L(h_{2}, k_{2}, b_{2}; R_{1,2})}\right) L(h_{2}, k_{2}, b_{2}; R_{1,2}) \quad (31)$$

and when $i = M$

$$T_{M} = \frac{D_{M} - L_{M} - F_{M}}{D_{M}} T_{M,M} + \frac{L_{M} + F_{M}}{D_{M}} T_{M,M-1} \quad (32)$$

with

$$L_{M} \approx L(h_{M-1}, k_{M-1}, b_{M-1}; R_{M,M-1}) \quad (33)$$

$$F_{M} \approx \left(\frac{D_{M}}{L(h_{M}, b_{M}; R_{M,M}) + L(h_{M-1}, k_{M-1}, b_{M-1}; R_{M,M-1})}\right) L(h_{M-1}, k_{M-1}, b_{M-1}; R_{M,M-1}). \quad (34)$$
The system throughput $T$ is thus given by

$$T = \sum_{i=1}^{M} \pi_i T_i. \quad (35)$$

Similarly, the system reliability is given by

$$P(E) = \sum_{i=1}^{M} \pi_i P_i(E) \quad (36)$$

where

$$P_i(E) = \frac{D_i - L_i - F_i}{D_i} P_{i,1}(E) + \frac{L_i + F_i}{D_i} P_{i,2}(E) \quad (37)$$

$$P_M(E) = \frac{D_M - L_M - F_M}{D_M} P_{M,M}(E) + \frac{L_M + F_M}{D_M} P_{M,M-1}(E) \quad (38)$$

$$P_i(E) = \frac{N_{i,i-1} P_{i,i-1}(E) + N_{i,i} P_{i,i}(E) + N_{i,i+1} P_{i,i+1}(E)}{D_i} \quad (39)$$

The performance of the $M$-state system is a function of $(3M - 2)$ test constants and can be optimized by maximizing the throughput under the constraint $P(E) \leq P^*$ where $P^*$ is the maximum allowed probability of error. This is a $(3M - 2)$-dimensional nonlinear optimization problem.

The performance expressions indicate that shorter ARL's for the composite tests after channel state changes are desirable. This is intuitively pleasing for it indicates that a system which reacts quickly to changes in the channel will yield better performance. Thus for channels with very long dwell times, this system performs almost as well as the ideal system equipped with instantaneous, perfect channel state information.

### 6 BCH Code Example

Consider a family of length $n$ BCH codes. Suppose in the family of codes there are $l$ codes from which to choose:

$$(n, k_1, d_1), (n, k_2, d_2), \ldots, (n, k_l, d_l)$$

where $d_i$ is the minimum distance of the $(n, k_i)$ for $i = 1, 2, \ldots, l$. Each $(n, k_i, d_i)$ code is allowed to correct up to $t$ errors. If $t = 0$ then the $(n, k_i, d_i)$ code is used as an ARQ error control code; if $0 < t < \lfloor \frac{d_i-1}{2} \rfloor$ then it used to perform hybrid-ARQ error control; and if $t = \lfloor \frac{d_i-1}{2} \rfloor$ it is used as an FEC code. The error control strategy $C_j$ channel state $j$ described
by $P_j(m, n)$ can be found by a simple computer search. For each $k_i$, the $\lfloor \frac{d-1}{2} \rfloor$ possible values of $t$ ($t = 0, 1, \ldots, \lfloor \frac{d-1}{2} \rfloor - 1$) define $\lfloor \frac{d-1}{2} \rfloor$ different ARQ or Hybrid-ARQ coding strategies. For each $(k_i, t)$ pair, the corresponding retransmission and undetected error probabilities, denoted $P_j(R; k_i, t)$ and $P_u(E; k_i, t)$, respectively, are evaluated. Next the throughput is determined. The $(k_i, t)$ pair for which the throughput is maximized while simultaneously maintaining $P_u(E; k_i, t) \leq P^*$ is chosen as the coding strategy for channel state $j$. Thus, using $T(k_i, n, P_j(R; k_i, t))$ to denote the general form for the throughput, the coding strategy for channel state $j$ denoted $C_j$ is described by the $(k^*, t^*)$ pair defined by

$$T(k^*, n, P_j(R; k^*, t^*)) = \max \left\{ T(k_i, n, P_j(R; k_i, t)) \mid P_u(E; k_i, t) \leq P^* \right\}.$$  \hspace{1cm} (40)

The use of length 127 BCH codes on the three state channel illustrated in Figure 8 is considered in this example. The use of BCH codes in a type-I hybrid-ARQ scheme has been considered before [18]. The decoding algorithm used in most applications is a bounded distance decoding algorithm that iterively generates an “error location polynomial” from the syndrome of the received word. The roots of this polynomial indicate the positions of the errors in the lowest weight error pattern associated with the syndrome. Thus the degree of the error locator polynomial can be used as a source of reliability information since it is a reliable estimate of the number of bit errors that have corrupted the received word. The modification of this decoder is an application of the modification to the more general non-binary decoder for Reed-Solomon codes developed by Wicker [19].

It is desired to maintain a probability of undetected error $\leq 10^{-6}$ and a selective repeat retransmission protocol [20] is assumed. Using equation (40) the error control strategies for

---

**Figure 8: Three State Channel of the BCH Code Example**

- $S_1$ is BSC, $\varepsilon = 2 \times 10^{-4}$
- $S_2$ is BSC, $\varepsilon = 10^{-3}$
- $S_3$ is BSC, $\varepsilon = 10^{-2}$
this example are

\[ C_1 = (127, 120) \quad t = 0 \]
\[ C_2 = (127, 113) \quad t = 1 \]
\[ C_3 = (127, 92) \quad t = 4. \]

Using a simplex optimization routine in 7 dimensions [21, 22], the test constants were determined. The tests \( Q_1, Q_2, \) and \( Q_3 \) used by the channel state monitor are

\[ Q_1 = \text{increasing test where} \quad h_1 = 39 \quad b_1 = 13 \]
\[ Q_2 = \text{two sided test where} \quad h_2 = 349 \quad k_2 = 214 \quad b_2 = 134 \]
\[ Q_3 = \text{decreasing test where} \quad k_3 = 314 \quad b_3 = 211. \]

The performance of the ideal channel state monitor over this channel is characterized by

\[ T_{\text{ideal}} = 0.8653 \]
\[ P_{u\text{ideal}}(E) = 5.7590 \times 10^{-8} \]

while the performance achieved by the adaptive system is

\[ T = 0.8622 \]
\[ P_u(E) = 8.5657 \times 10^{-7}. \]

The "worst case" channel is characterized by a crossover probability of \( 10^{-2} \) and requires the error control of \( C_3 \). The average channel crossover probability is \( 2.48 \times 10^{-3} \) and would require the \((127, 106)\) \( t = 2 \) code to optimize the performance if the channel were treated as a one state channel with the average channel crossover probability. The performance of these coding strategies is summarized in Table 1.

Here the throughput penalty for using sequential tests to monitor the channel is a mere 0.36%. The adaptive system offers a 16% improvement in throughput over the worst case scheme and a 6% throughput improvement over the coding strategy designed for the average channel conditions. Even though the improvement over the average channel coding strategy appears small, it should be noted that the average channel coding strategy does not offer a probability of undetected error \( \leq P^* = 10^{-6} \). As expected, the worst case scheme offers the best reliability and the lowest throughput. In this case the adaptive system can improve throughput performance by as much as 16% without a serious degradation in reliability. This improvement however is accompanied by an increase in the complexity of the

22
Table 1: Performance Summary and Comparisons for the Length 127 BCH Codes over a Three State Channel

<table>
<thead>
<tr>
<th>Coding Strategy</th>
<th>$T$</th>
<th>$P_u(E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ideal</td>
<td>0.8653</td>
<td>$5.7590 \times 10^{-8}$</td>
</tr>
<tr>
<td>adaptive</td>
<td>0.8622</td>
<td>$8.5657 \times 10^{-7}$</td>
</tr>
<tr>
<td>worst case</td>
<td>0.7231</td>
<td>$1.9708 \times 10^{-8}$</td>
</tr>
<tr>
<td>average channel</td>
<td>0.8120</td>
<td>$8.3423 \times 10^{-6}$</td>
</tr>
</tbody>
</table>

encoder/decoder system. It must be able to encode and decode three different BCH codes. It is evident that the use of adaptivity incorporating channel state information improves the performance.

7 Convolutional Code Example

The family of rate compatible punctured convolutional (RCPC) codes developed by Kallel and Haccoun [23] and Hagenauer [24] can be applied to the adaptive error control scheme based on type-I hybrid-ARQ protocols.

The modification proposed by Yamamoto and Itoh [25] to the maximum likelihood decoder using the Viterbi algorithm is used as the decoding rule in this example. This modification is described as follows: Assume BPSK modulation over the AWGN channel with one-sided spectral density $N_0$ (W/Hz). Thus each channel symbol $x^m \in \{-1,+1\}$ and an energy $E$ Joules is transmitted for each symbol. The optimum receiver demodulates the received signal by a single phase coherent correlation demodulator which has normalized output $y_j$ given by

$$y_j = \sqrt{\frac{2E}{N_0}} x^m_j + n_j$$

where $n_j$ is a zero mean unit variance Gaussian random variable. It has been shown that the inner product

$$y_j \cdot x^m_j = \sum_{i=1}^{j} y_i x^m_i$$

serves as a sufficient measure of the metric in place of the log likelihood function when the channel is memoryless [26]. At each node of level $j$, the decoder selects the paths $x^m_j$ and
that have the largest and second largest inner products, respectively. If at any level the condition

\[ y_j \cdot x_j^n - y_j \cdot x_j^{n'} < u \]  

(41)
is satisfied for some nonzero real number \( u \), the surviving path is labeled unreliable\(^3\). A retransmission is requested when all surviving paths have been declared unreliable.

Each member of the family of RCPC codes is defined by its perforation matrix \( P_r \) and its rate \((V-1)/(V+rh)\) where \( h \) is the number of symbols punctured from each member of the family to generate the next higher rate member and \( V \) is the puncturing period. In this sense, \( r \) can be used as an index for the family of codes (i.e. knowledge of the rate \( 1/n_0 \) mother code, \( h \), and \( V \) permits one to uniquely specify each member of the family by \( r \)). For each member of the family of RCPC codes the smallest value of \( u \) that yields \( P_b(E) < P* \) on channel state \( S_j \) is determined. Next the throughput is calculated for the \((u,r)\) pair. The \((u,r)\) pair for which the throughput is maximized under the constraint \( P_b(E) < P* \) defines the coding strategy \( C_j \). To be precise, over channel state \( j \) let \( T_j(u,r) \) represent the throughput for the member of the family indexed by \( r \) and let \( P_b(E,u,r) \) represent the probability of bit error for the member indexed by \( r \). The arguments \( u \) and \( r \) are used to emphasize the functional dependence of the performance on those parameters in this formulation. Then for channel state \( j \), \( C_j \) is defined by the pair \((u^*,r^*)\) where

\[
T_j(u^*,r^*) = \max_r \{ T_j(u^*(r),r) \}
\]

(42)

for \( 0 \leq r \leq \left\lfloor \frac{(n_0-1)(V-1)-1}{h} \right\rfloor \).

\[
u^*(r) = \min \{ u \geq 0 \mid P_b(E,u,r) \leq P* \}\]

(43)

The family of convolutional codes defined by \( V = 8, n_0 = 2, h = 1, \) and \( m = 6 \) (the best rate \( 7/8 \) code punctured from the best \( (2,1,6) \) convolutional code) listed in [23] is applied to the three state AWGN channel shown in Figure 9. It is desired to maintain a bit error rate \( \leq 10^{-5} \) on this channel. With a packet length of \( N = 1000 \) and using equations (42) and (43), the coding strategies for the three channel states assuming a selective repeat retransmission protocol [20] are

\[
C_1 = (8,7,6) \quad u = 0.8
\]

\[
C_2 = (10,7,6) \quad u = 0.8
\]

\[
C_3 = (13,7,6) \quad u = 1.5.
\]

\(^3\)Note that when \( u = 0 \) this rule corresponds to the traditional FEC Viterbi decoding rule.
Using a simplex optimization routine in 7 dimensions [21, 22], the test constants were determined. The tests \( Q_1, Q_2, \) and \( Q_3 \) used by the channel state monitor are

\[
\begin{align*}
Q_1 &= \text{increasing test where } h_1 = 15 \quad b_1 = 7 \\
Q_2 &= \text{two sided test where } h_2 = 490 \quad k_2 = 722 \quad b_2 = 489 \\
Q_3 &= \text{decreasing test where } k_3 = 289 \quad b_3 = 253.
\end{align*}
\]

The performance of the ideal channel state monitor over this channel is characterized by

\[
T_{\text{ideal}} = 0.7218 \quad \quad P_{b_{\text{ideal}}}(E) = 9.4293 \times 10^{-6}
\]

while the performance achieved by the adaptive system is

\[
T = 0.7199 \quad \quad P_{b}(E) = 9.9938 \times 10^{-6}.
\]

The "worst case" channel is characterized by a SNR = 1 dB and requires the error control provided by \( C_3 \). The average SNR is 3.3 dB and would require the (13, 7, 6) \( u = 1.5 \) code to optimize the performance if the channel were treated as a one state channel with the average SNR. The performance of these coding strategies on the three state channel is summarized in Table 2.

Here the throughput of the adaptive system compares quite favorably with that of the ideal system. The penalty for using sequential tests to monitor the channel is 0.26%. The
Table 2: Performance Summary and Comparisons for the Family of Convolutional Over the Three State Channel

<table>
<thead>
<tr>
<th>Coding Strategy</th>
<th>$T$</th>
<th>$P_b(E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ideal</td>
<td>0.7218</td>
<td>$9.4293 \times 10^{-6}$</td>
</tr>
<tr>
<td>adaptive</td>
<td>0.7199</td>
<td>$9.9938 \times 10^{-6}$</td>
</tr>
<tr>
<td>worst case</td>
<td>0.5731</td>
<td>$1.3837 \times 10^{-6}$</td>
</tr>
<tr>
<td>average channel</td>
<td>0.6200</td>
<td>$2.8473 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

The adaptive system offers a 20% improvement in throughput over the worst case scheme and a 14% throughput improvement over the coding strategy designed for the average channel conditions. Note here that the average channel coding strategy does not offer a decoded bit error rate $\leq P^* = 10^{-5}$. As expected, the worst case scheme offers the best reliability and the lowest throughput. Compared with nonadaptive schemes, the adaptive scheme offers a substantial improvement in throughput by as much as 20% without a tremendous increase in the probability of bit error.

8 Conclusions

The adaptive system for error control over slowly varying channels has been shown to be an effective method for improving performance. When compared to conventional nonadaptive coding strategies, this method significantly improves throughput without a serious degradation in reliability. This increase in performance however is realized at the expense of increased complexity. Expressions approximating the performance of this scheme were derived based on the average run length of the inspections schemes used as channel monitors.

The application of this concept to systems using BCH and convolutional codes with modified Viterbi decoding were presented. This method could also be applied to other block decoders such as the majority logic decoder and to sequential decoding modified by the slope control algorithm[27] or the time out algorithm[28]. Further, the sequential approach to monitoring the channel overcomes the difficulties inherent in the "blocked" approach used by those schemes found in the literature. The sequential test in this application exhibits a shorter response time thereby making it ideal for use in such a system.
References


Packet Combining Systems
Based on the Viterbi Decoder

Bruce A. Harvey and Stephen B. Wicker
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

December 8, 1991

This work was supported by the National Science Foundation under grant number NCR-9009877
Abstract

Type-I hybrid-ARQ protocols can be used to construct powerful adaptive rate algorithms through the use of packet combining techniques. In this paper several packet combining schemes are presented for use in conjunction with the Viterbi decoder over stationary and time-varying channels. The first technique presented is an averaged diversity combiner, which is shown to be identical in performance to an interleaved code combiner over an AWGN channel. The averaged diversity combiner is then generalized to make use of packet weights based on either ideal channel state information or weights derived from side information generated by the Viterbi decoder. It is shown that the weighted diversity combiner using decoder side information performs almost as well as the system using ideal channel state information. All of the packet combining schemes discussed in this paper provide improved throughput and reliability performance relative to that provided by the standard type-I hybrid-ARQ or FEC systems. This performance improvement is obtained at the expense of negligible to moderate modifications to the transmitter and receiver. Performance bounds are derived for each of the combining schemes and their tightness verified through simulation results.
1 Introduction

Many communication channels exhibit widely varying noise and attenuation levels. These variations may be due to changes in the weather, the number of active users on the channel, or any of a number of other channel parameters. In all cases, however, the error control system designer is presented with a formidable problem. He or she may proceed under the assumption that the channel will exhibit its worst behavior at all times. The resulting error control system provides excellent reliability performance, but the throughput performance is poor due to the transmission of unnecessary redundancy when the channel conditions are relatively good. The ideal solution is an adaptive system that carefully matches the rate of the error control code to the channel conditions.

The type-II hybrid-ARQ protocol [1,2,3,4,5] can be viewed as a simple adaptive rate system. As channel conditions vary, the instantaneous code rate of the type-II system varies between a base rate R and a lower rate R/2. If the channel can be modeled as a two-state Markov process (each state corresponding to a stationary channel), then the type-II system provides an excellent error control solution. If, on the other hand, the channel tends to exhibit three or more distinct noise/attenuation levels, or even a wide continuum of such levels, a more complex approach must be taken to optimize throughput and reliability.

In any communication system that uses a feedback channel for retransmission requests, throughput and reliability performance can be improved through the use of packet combining. In packet combining systems the receiver forms increasingly reliable estimates of the transmitted data by combining all received packets prior to decoding, improving the reliability of the decoded data and decreasing the probability of further retransmission requests. Through the combination of an arbitrary number of packets, the error control system is able to effect multiple code rates, providing a higher level of adaptivity to changing channel conditions than the basic type-II hybrid-ARQ protocol. Benelli [6,7], Bruneel and Moeneclaey [8], Lau and Leung [9], Metzner and Chang [10], Sindhu [11], and Wicker [12] have proposed methods for combining packets using diversity (symbol-
by-symbol) combining. Chase [13], Krishna, Morgera, and Odoul [14,15], Hagenauer [2], Kallel and Haccoun [3,16,17], and Wicker [18,19] have proposed methods for combining packets using code (codeword-by-codeword) combining.

Hagenauer [2] was the first to note that a type-II hybrid-ARQ protocol based on rate-compatible punctured convolutional (RCPC) codes could be improved through the application of the code combining techniques developed by Chase [13]. Kallel and Haccoun [3,16] have explored this idea in detail by interleaving multiple received convolutionally encoded packets to form a single code word from a convolutional repetition code. In this paper the approach of Kallel and Haccoun is modified and then generalized for use over time-varying channels. The modification consists of the use of the Yamamoto and Itoh algorithm [22] for error detection in the Viterbi decoder as opposed to the use of CRC error detection after Viterbi error correction. The modified Viterbi decoder implements a type-I hybrid-ARQ protocol as defined by Lin, Costello, and Miller [5]. A brief review of the Yamamoto and Itoh algorithm and its performance is provided in Section 2.

In many cases it is possible to use combinatorial techniques in conjunction with the subject code's weight enumerator to obtain exact performance expressions for packet combining systems (for example, this is the case with systems based on MDS codes [18,19]). In convolutionally encoded systems, however, this approach is impracticable for much the same reason that the performance of convolutionally encoded FEC systems is discussed in terms of upper and lower bounds: the union bound cannot be avoided in the analysis [23] without an excursion into an intractable series of combinatorial exercises. In Section 3 the Kallel-Haccoun performance bounds for packet combining systems [3,16,25] are reviewed and extended. This paper makes extensive use of the Kallel-Haccoun bounds, and through simulation results, shows them to be quite tight in the examples treated here.

In Section 4 several packet combining techniques based on the Viterbi decoder are presented and analyzed. The first system is a simple averaged diversity combiner (ADC) in which the average soft decision value is obtained for all received copies of each bit in the received packet and the resulting combined packet decoded. The ADC is shown here to
provide exactly the same performance as the interleaved code combining system of Kallel and Haccoun [16,17] over an AWGN channel. The ADC system is then generalized for use with time-varying channels through the use of packet weighting schemes. The first scheme uses ideal channel information to create weights in the manner suggested by Chase [13]. The second scheme derives packet weights from side information generated by the Viterbi decoder. It is shown in Section 5 that the latter scheme performs almost as well as the former.

In Section 5 the performance bounds for the various packet combining schemes discussed in this paper are corroborated through the use of simulation results. In all cases it is shown that the use of packet combining substantially improves throughput performance in comparison to the type-I hybrid-ARQ protocol based on the Viterbi decoder.

2 Type-I Hybrid-ARQ Viterbi Decoders

The key to modifying an FEC decoder for use in a type-I hybrid-ARQ protocol is the identification of a source of reliability information within the decoding process [20,21]. This information is used to estimate the reliability of received packets, indicating whether a retransmission request is in order. The path metrics calculated during Viterbi decoding provide just such a source of information. In the Yamamoto-Itoh algorithm [22], the surviving path and the best non-surviving path are compared at each node at each stage in the decoding process. If the difference in the path metrics of the two paths falls below a threshold \( u \), then the survivor is declared unreliable. If all paths are declared unreliable before decoding is completed, then a retransmission request is generated.

As noted in the introduction, the use of the Yamamoto-Itoh algorithm for error detection in a packet combining system is a departure from the use of CRC error detection as exhibited previously in the literature (for example, see [3,16]). There are two benefits that arise from the use of the Yamamoto-Itoh algorithm: first, there is no longer any need for the CRC encoder/decoder pair. Though CRC systems are extremely simple to implement,
they do introduce significant delay. Second, the Yamamoto-Itoh algorithm allows for the
dynamic reallocation of error correction and error detection through control of the param-
eter \( u \). The CRC approach does, however, offer the advantage of greater flexibility in the
initial selection of an error detection system (and therefore its performance) independent
of the error correction capacity of the code combining system.

The performance analysis for the Yamamoto-Itoh algorithm using soft decision decoding
follows a method developed by Viterbi [23]. It is assumed that binary code symbols are
transmitted by binary phase-shift keyed (PSK) modulation over an AWGN channel with
one-sided noise spectral density \( N_o \) (W/Hz). Each symbol is transmitted with an energy
of \( E \) (Joules) and the receiver demodulates the received signal using a phase coherent
demodulator. The normalized demodulator output \( y_t \) is given by \( y_t = x_t + n_t \), where
\( x_t \in \{+1, -1\} \) and \( n_t \) is a zero mean Gaussian random variable with variance \( \frac{N_o}{2E} \).

The probability of retransmission is upper bounded by [24]

\[
P_x = 1 - (1 - P_{x1})^\frac{N}{n},
\]

where

\[
P_{x1} \leq \sum_{i=d_{free}}^\infty a_i P_x(i),
\]

\[
P_x(i) \leq Q\left[\frac{2i - u}{2\sqrt{i\frac{N_o}{2E}}}\right],
\]

\( a_i \) is the number of error paths of weight \( i \) for the convolutional code used, \( n \) is the number
of output bits per code trellis branch, \( N \) is the length of the transmitted packet, and \( u \) is
the retransmission threshold.

The decoded bit error rate for the Yamamoto-Itoh algorithm is bounded above by [24]

\[
P_b \leq \frac{1}{m} \sum_{i=d_{free}}^\infty c_i P_x(i)
\]

\(^1\text{Note that in the derivation of the soft decision performance bounds provided by Yamamoto and Itoh in}
\[22\], the demodulator output is normalized such that the random component \( n_t \) has unit variance.
where
\[ P_e(i) = Q \left( \frac{2i + u}{2\sqrt{\frac{i}{2^E}} \frac{N}{2}} \right), \]

\( m \) is the number of information bits per code trellis branch, and \( c_i \) is the total number of nonzero information bits associated with all code words of weight \( i \).

3 Performance Bounds for Packet Combining Systems

One of the principle difficulties encountered in the analysis of code combining schemes is that packets that cause the generation of retransmission requests are "noisier" on the average than those that have not. Kallel and Haccoun have developed a series of expressions that characterize this phenomenon [3,16,25,17].

Let \( T_r \) be the expected number of transmission attempts that must be made before a packet is accepted by the receiver (whether correctly or incorrectly). Kallel and Haccoun showed that \( T_r \) is bounded above and below by

\[ 1 + \sum_{L=1}^{\infty} \prod_{j=1}^{L} P(R_j) \leq T_r \leq 1 + \sum_{L=1}^{\infty} P(R_L), \]

(6)

where \( P(R_L) \) is the probability of generating a retransmission request while decoding the packet formed by combining \( L \) received copies of the packet. Bounds on the expected throughput \( T \) of the adaptive rate system using an ideal selective repeat protocol are then obtained through the expression \( T = \frac{R}{T_r} \), where \( R \) is the code rate.

Upper and lower bounds on the decoded bit error rate \( P_B \) provided by a packet combining system can be derived in a similar manner [24]:

\[ P_B \leq P(B_1) + \sum_{L=1}^{\infty} \left( \prod_{j=1}^{L} P(R_j) \right) [P(B_{L+1}) - P(B_L)] \]

(7)

and

\[ P_B \geq P(B_1) + \sum_{L=1}^{\infty} P(R_L)[P(B_{L+1}) - P(B_L)], \]

(8)
where $P(B_L)$ is the probability of bit error in the decoded data after $L$ received copies are combined and the result accepted by the decoder (i.e. there are no further retransmission requests). In the following analyses the principal objective is the derivation of values for $P(R_L)$ and $P(B_L)$. Once these values are known, the above expressions can be used to bound the overall throughput and reliability performance for the various packet combining Viterbi decoders.

4 Packet Combining Systems for the Viterbi Decoder

In this section several packet combining techniques are presented and analyzed. These techniques are actually successive generalizations of the same basic idea. The first technique discussed is an averaged diversity combiner (ADC) that combines packets bit-by-bit by averaging their soft-decision values. It is shown that this technique is identical to the interleaved code combining technique presented by Kallel and Haccoun [3,16] when both techniques are used over an AWGN channel. The ADC is generalized for use with time-varying channels through the addition of packet-by-packet weighting. The first weighting scheme uses ideal channel information in the manner suggested by Chase [13]. The second scheme uses weights derived from side information generated by the Viterbi decoder. It is shown later in this paper that the two weighting schemes provide almost identical results.

The packet combining systems presented here are analyzed for both stationary and time-varying channels. In the stationary and very slowly-varying cases, it is assumed that the channel noise and attenuation level is constant during the transmission of all packets involved in a given packet combining operation. As packet weighting is superfluous in this case, analytical and simulation results are only provided for the ADC. A geosynchronous satellite channel provides a good example of a slowly-varying channel. Weather conditions cause long-term changes in the channel, but do not occur fast enough to cause a significant difference between the packets in a given combining operation. The ADC adapts to long-term changes in the channel by varying the number of packets combined prior to accepting
the results of a decoding operation. As will be shown in the examples in Section 5, the ADC thus extends the operating range over which an acceptable level of throughput can be obtained in comparison to the basic type-I hybrid-ARQ protocol.

In the case of the "moderately-varying" channel it is assumed that the channel noise and attenuation level is constant during the transmission of a packet, but may vary from packet to packet. This typically occurs in channels in which the propagation delay is long compared to the transmission time (e.g. some meteor-burst channels). Bursty channels can also be placed under this heading, for a burst may affect one packet during its transmission without affecting the next. In either case retransmitted packets may encounter a substantially different channel than that seen by the initial transmission. Moderately-varying channels are to be characterized as follows.

Channel conditions are described by the linear SNR with probability density function (pdf) $f_{SNR}$, the SNR in decibels $SNR_{dB}$ with pdf $f_{dB}$, the variance $\sigma^2$ with PDF $f_{\sigma^2}$, or the channel bit error rate $p$ with pdf $f_p$. The effective channel pdf $f_{U,L}(x)$ shall be used to characterize the noise corrupting the symbols in a packet formed through the combination of $L$ received packets. The units of $x$ are indicated by the subscript $U$, where

$$U \in \{SNR, dB, \sigma^2, p\}. \quad (9)$$

$f_{U,L}(x)$ is derived for each combining method presented in this section.

The probability of packet acceptance on a given transmission as a function of the noise on the channel is given as $P_{A,U}(x)$. The probability of decoded bit error as a function of $x$ given that the packet is accepted is similarly given by $P_{B,U}(x)$. $P_{A,U}(x)$ and $P_{B,U}(x)$ are functions of the performance of the Yamamoto-Itoh algorithm.

The probability that a packet is accepted after exactly $L$ copies of the packet have been received is seen to be

$$P(A_L) = \int_{-\infty}^{\infty} f_{U,L}(x)P_{A,U}(x)dx. \quad (10)$$

The probability of decoded bit error given that $L$ copies of the packet have been received
and the packet is accepted is

\[ P(B_L) = \int_{-\infty}^{\infty} P_{B,U}(x) \text{Prob.}(x|L \text{ copies received, packet accepted}) dx \]  

(11)

where

\[ \text{Prob.}(x|L \text{ copies received, packet accepted}) = \frac{f_{U,L}(x) \text{Prob.}(\text{Accepted}|x)}{\int_{-\infty}^{\infty} f_{U,L}(x) P_{A,U}(x) dx} \]

\[ = \frac{f_{U,L}(x) P_{A,U}(x)}{P(A_L)}. \]  

(12)

The probability of bit error in the decoded packet is thus

\[ P(B_L) = \frac{1}{P(A_L)} \int_{-\infty}^{\infty} P_{B,U}(x) f_{U,L}(x) P_{A,U}(x) dx. \]  

(13)

In the following subsections, the effective probability density functions for the averaged diversity combining scheme and the weighted diversity combining schemes are derived. Equations (10) and (13) are then used to calculate \( P(R_L) = 1 - P(A_L) \) and \( P(B_L) \). System performance bounds are then calculated using the general adaptive rate bounds.

### 4.1 Interleaved Code Combining and Averaged Diversity Combining

Interleaved code combining is, as its name implies, a code combining method where the symbols in the received copies of a packet are interleaved to form a single packet at the receiver. For example suppose that \( L \) copies of a packet have been received. Let \( V_i = (y_{i1}y_{i2}y_{i3} \cdots y_{iN}) \) be the \( i^{th} \) received packet, where \( y_{ij} \) is the \( j^{th} \) bit of the \( i^{th} \) packet. The combined packet is

\[ Y_i = (y_{11}y_{21} \cdots y_{L1}y_{12}y_{22} \cdots y_{L2} \cdots y_{1k}y_{2k} \cdots y_{LN}), \]

a noise corrupted code word from a convolutional repetition code. If the original encoder has a code rate \( R \) and minimum free distance \( d_{\text{free}} \), then the combined packet has an effective code rate of \( R/L \) and minimum free distance of \( Ld_{\text{free}} [2] \).
Averaged diversity combining reflects a different approach to packet combining. Suppose that \( L \) packets have been received. The \( L \) packets are combined into a single packet of the same length by averaging the soft decision values of the copies of each code word coordinate. If \( V_i = (y_{i1} y_{i2} y_{i3} \ldots y_{iN}) \) is the \( i \)th received packet, the new packet is \( Y_D = (z_1 z_2 \ldots z_N) \), where

\[
z_j = \frac{1}{L} \sum_{i=1}^{L} y_{ij}.
\]

The decoder itself need not be modified beyond the type-I hybrid-ARQ modification to be able to decode the combined packet, for the combined packet is a noise corrupted code word from the same code used for each of the individual packet transmissions.

Given a received packet corrupted by transmission over an AWGN channel, an interleaved code combining Viterbi decoder and an averaged diversity combining Viterbi decoder always select the same code word. This can be shown through a comparison of the path metrics calculated by the Viterbi decoder for each of the combining methods. Assume that the convolutional code has rate \( \frac{m}{n} \) and that the individual noise processes affecting the received symbols are statistically independent. Without code or diversity combining, the partial path metric for the \( t \)th trellis branch is

\[
M \propto \sum_{j=(t-1)n+1}^{t_n} \log p(y_j|x_j),
\]

where \( y_j \) is the \( j \)th received symbol and \( x_j \) is the \( j \)th transmitted code symbol. Assume that \( L \) code words are combined using interleaved code combining. If \( y_{ij} \) is the \( j \)th received symbol from the \( i \)th received packet, then the partial path metric for the interleaved word over the \( t \)th transition is

\[
M_I \propto \sum_{i=1}^{L} \sum_{j=(t-1)n+1}^{t_n} \log p(y_{ij}|x_j).
\]

If the channel is AWGN, the inner product provides the proper metric [23], giving the following branch metric for the interleaved code combiner.

\[
M_I = \sum_{i=1}^{L} \sum_{j=(t-1)n+1}^{t_n} y_{ij} x_j
\]
The symbols for the new code word formed by averaged diversity combining are

\[ z_j = \frac{1}{L} \sum_{i=1}^{L} y_{ij}. \]  

(18)

The partial path metric over the \( t^{th} \) transition is then

\[ M_A \propto \sum_{j=(t-1)n+1}^{tn} \log p(z_j|x_j). \]  

(19)

Assuming an AWGN channel, it is then clear that

\[ M_A = \sum_{j=(t-1)n+1}^{tn} z_j x_j = \sum_{j=(t-1)n+1}^{tn} \frac{1}{L} \sum_{i=1}^{L} y_{ij} x_j = \frac{1}{L} \sum_{i=1}^{L} \sum_{j=(t-1)n+1}^{tn} y_{ij} x_j = \frac{1}{L} M_I. \]  

(20)

Since the branch metrics for the interleaved code combiner and the averaged diversity combiner always differ by a fixed factor \( L \), the decisions made at the nodes of the trellis diagram are identical. The Viterbi decoder chooses the same code word regardless of which combining method is used.

The ADC and the interleaved code combiner thus provide the same performance over an AWGN channel. It is important to note, however, that these two techniques reflect two different approaches to packet combining. In general, code combining systems offer better throughput and reliability performance than diversity combining systems (as noted by Chase [13]) while diversity combining systems are easier to implement. In this particular instance the two approaches converge.

The ADC system is implemented here using a Viterbi decoder that incorporates the Yamamoto-Itoh algorithm. Equations (1) and (4) provide the probability \( P_b \) of bit error in the decoded packets and the probability \( P_c \) that retransmission request is generated.

The variance of a symbol resulting from the combination of \( L \) symbols by the averaged diversity combiner is

\[ \sigma^2(L) = \frac{N_o}{2EL} = \frac{\sigma^2}{L}, \]  

(21)

where \( \sigma^2 \) is the variance of the noise process corrupting the individual demodulated signals before combining. The probability of bit error for a packet that has been accepted after \( L \)
copies have been combined is then
\[ P(B_L) \leq \frac{1}{m} \sum_{i=d_{free}}^{\infty} c_i Q \left( \frac{2i + u}{2\sqrt{i^2 L}} \right). \tag{22} \]

The probability of a retransmission request after \( L \) packet have been combined is
\[ P(R_L) = P_z(L) = 1 - [1 - P_{z1}(L)]^n, \tag{23} \]
where
\[ P_{z1}(L) \leq \sum_{i=d_{free}}^{\infty} a_i Q \left( \frac{2i - u}{2\sqrt{i^2 L}} \right). \tag{24} \]

Upper bounds on the expected number of transmissions and the probability of decoded bit error for the averaged diversity combiner system can now be calculated using the bounds for a general adaptive rate system in equations (6) and (7).

To derive performance bounds for the ADC system over a non-stationary channel it is necessary to find the effective pdf \( f_{X,L} \) for the symbols in the combined packet, the probability of packet acceptance \( P_{A,X}(x) \), and the probability of decoded bit error \( P_{B,X}(x) \) as a function of the channel noise variance \( X = \sigma^2 \).

Let \( X = \{x_1, x_2, x_3, \ldots, x_N\} \), where \( x_i \in \{1, -1\} \), be the packet that the transmitter is attempting to send over the channel. Assume that \( L \) packets \( Y_1, Y_2, Y_3, \ldots, Y_L \) have been received with corresponding noise variances \( \sigma_1^2, \sigma_2^2, \sigma_3^2, \ldots, \sigma_L^2 \). Let \( y_{ij} \) be the demodulator output resulting from the reception of the \( j^{th} \) bit of the \( i^{th} \) received packet. The combined packet \( Z = \{z_1, z_2, z_3, \ldots, z_N\} \) is formed using
\[ z_j = \frac{1}{L} \sum_{i=1}^{L} y_{ij}. \tag{25} \]

The expected value of \( z_j \) is \( E\{z_j|x_j\} = x_j \) and the variance of \( z_j \) is
\[ V\{z_j\} = \frac{1}{L^2} \sum_{i=1}^{L} \sigma_i^2. \tag{26} \]

Let \( f_{\sigma^2}(x) \) be the pdf of the noise variance of the channel. The pdf of the variance of the combined packet \( V\{z\} \) is
\[ f_{\sigma^2,L}(x) = f_{\text{sum},L}(xL^2), \tag{27} \]
where
\[ f_{\text{sum},L}(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \Phi^L(w)e^{-jwx}dw \] (28)
is the pdf of the sum of the L variances and
\[ \Phi(w) = \int_{-\infty}^{\infty} f_{\sigma^2}(x)e^{jwx}dx \] (29)
is the characteristic function of the channel noise variance.

The functions \( P_{A,\sigma^2}(x) \) and \( P_{B,\sigma^2}(x) \), where \( x \) is in units of noise variance, are also needed to derive the performance bounds. Using the bounds on the performance of the Yamamoto and Itoh algorithm, these functions are found to be
\[ P_{B,\sigma^2}(x) \leq \frac{1}{m} \sum_{i=d_{\text{free}}}^{\infty} c_i Q \left[ \frac{2i + u}{2\sqrt{ix}} \right] \] (30)
\[ P_{A,\sigma^2}(x) = [1 - P_{x1}(x)]^\frac{u}{2}, \] (31)
where
\[ P_{x1}(x) \leq \sum_{i=d_{\text{free}}}^{\infty} a_i Q \left[ \frac{2i - u}{2\sqrt{ix}} \right], \] (32)
and the coefficients \( a_i \) and \( c_i \) are derived from the generating function of the convolutional code.

Upper bounds on the expected number of transmissions and the probability of decoded bit error are now derived using equations (10), (13), and the general adaptive rate bounds.

4.2 Reliability Weighting Using Perfect Side Information

Suppose that perfect channel information is available during the transmission of each packet. For a binary symmetric channel (BSC) the desired information is the channel bit error rate \( p \). For an additive white Gaussian noise (AWGN) channel the desired information is the signal-to-noise ratio SNR, the signal-to-noise ratio in dB SNR\_dB, or the channel noise variance \( \sigma^2 \). Using this information, weighting factors can be derived for the maximum likelihood decoding of multiple packets [13].
For an AWGN channel the weighting factors depend on the variance of the noise during the transmission of each packet. Assume that $L$ copies of a packet of length $N$ have been received. Chase showed that

$$\log p(Y|X_m) \propto \sum_{i=1}^{L} \frac{1}{\sigma_i^2} \sum_{j=1}^{N} y_{ij} x_{m_j},$$

(33)

where $x_{m_j} = \pm 1$, $y_{ij} = a_j + n_{ij}$, $a_j = \pm 1$ is the transmitted bit, and $n_{ij}$ is a Gaussian random variable with variance $\sigma_i^2$ [13].

To obtain the weights for a diversity combining system one need only interchange the summations in the expression above and include the factor $\frac{1}{L}$.

$$\log p(Y|X_m) \propto \sum_{j=1}^{N} x_{m_j} \left[ \frac{1}{L} \sum_{i=1}^{L} \frac{1}{\sigma_i^2} y_{ij} \right]$$

(34)

It should be noted that even if hard decision reception is employed on each packet, a soft decision Viterbi decoder must be used to implement weighted diversity combining. The performance analysis is thus pursued only for the soft decision AWGN channel case\(^2\).

The performance analysis for the weighted diversity combiner is very similar to the analysis of the averaged diversity combiner. The combined packet $Z = \{z_1, z_2, z_3, \ldots, z_N\}$ is formed using the following expression:

$$z_j = \left[ \frac{1}{\sum_{i=1}^{L} \sigma_i^2} \right]^{-1} \sum_{i=1}^{L} \frac{1}{\sigma_i^2} y_{ij}.$$  

(35)

The conditional mean of $z_j$ is $E\{z_j|x_j\} = x_j$ where $x_j = \pm 1$ is the transmitted symbol. The variance of the combined symbols is

$$V\{z_j\} = \sigma_z^2 = \left[ \sum_{i=1}^{L} \frac{1}{\sigma_i^2} \right]^{-1}.$$  

(36)

By letting $\text{SNR}_i = 1/(2\sigma_i^2)$ be the signal-to-noise ratio associated with the $i^{th}$ copy of the packet, the effective signal-to-noise ratio of the combined packet $\text{SNR}_Z$ becomes

$$\text{SNR}_Z = \sum_{i=1}^{L} \text{SNR}_i.$$  

(37)

\(^2\)The performance of the BSC case with weighted diversity combining can be estimated from the AWGN channel with weighted diversity combining by applying a 2 dB [31] or a 3 dB [4] loss in SNR.
Let \( f_{\text{SNR}}(x) \) be the pdf of the signal-to-noise ratio associated with a single received packet. The pdf of the effective signal-to-noise of a packet formed through the combination of \( L \) packets is then

\[
 f_{\text{SNR}_z,L}(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \Phi(w)e^{-jwx}dw,
\]

where

\[
 \Phi(w) = \int_{-\infty}^{\infty} f_{\text{SNR}}(x)e^{jwx}dx
\]

is the characteristic function of the signal-to-noise ratio of the channel.

The probability of packet acceptance after \( L \) copies of the packet have been received \( P_{A,\text{SNR}}(x) \) and the probability of bit error \( P_{B,\text{SNR}}(x) \) are easily found from the Yamamoto-Itoh performance bounds. These are

\[
P_{B,\text{SNR}}(x) \leq \frac{1}{m} \sum_{i=d_{\text{free}}}^{\infty} c_i Q \left[ \frac{2i + u}{2\sqrt{i^2 + 2x}} \right],
\]

\[
P_{A,\text{SNR}}(x) = [1 - P_{x1}(x)]^N
\]

where

\[
P_{x1}(x) \leq \sum_{i=d_{\text{free}}}^{\infty} a_i Q \left[ \frac{2i - u}{2\sqrt{i^2 + 2x}} \right],
\]

and the coefficients \( a_i \) and \( c_i \) are derived from the generating function of the selected convolutional code.

Upper bounds on the expected number of transmissions and the probability of decoded bit error are now derived using equations (10), (13), and the general adaptive rate bounds.

### 4.3 Reliability Weighting Using Decoder Side Information

In the previous section the channel condition during the transmission of each packet was provided by a "perfect" outside source. In this section a method of estimating channel conditions using information generated by a soft decision Viterbi decoder is proposed and analyzed.

Assume that the \( i^{\text{th}} \) received packet \( Y_i = \{y_{i1}, y_{i2}, \ldots, y_{iN}\} \) was transmitted with a signal-to-noise ratio \( \text{SNR}_i \). This is equivalent to a noise variance of \( \sigma_i^2 = 1/(2\text{SNR}_i) \).
Assume also that the packet was decoded correctly by the Viterbi decoder (no combining at this point). The probability that this assumption is correct is $1 - P_e$, where $P_e$ is the probability of decoder error. For a rate $m/n$ convolutional code, each transition of the Viterbi trellis corresponds to $n$ coded bits and $m$ information bits. The partial path metric of the correct path over the $k^{th}$ transition of the Viterbi trellis is

$$ M_k = \sum_{j=(k-1)n+1}^{kn} x_j \cdot y_{ij} $$

where $X = \{x_1, x_2, \ldots, x_N\}$ is the transmitted packet, $x_i \in \{+1, -1\}$, $y_{ij} = x_j + n_{ij}$, and $n_{ij}$ is a zero-mean Gaussian random variable with variance $\sigma^2_i$. $M_k$ is thus the sum of $n$ Gaussian random variables with unit mean and variance $\sigma^2_i$ and thus has mean $n$ and variance $n\sigma^2_i$.

An estimate of the noise variance on the channel during the transmission of the packet $X_i$ is found by summing the values $(M_k - n)^2$ for $k = 1, 2, \ldots, N/n$ and dividing the result by $N$. To see this, consider the function

$$ \Omega = \frac{\sum_{k=1}^{N/n}(M_k - n)^2}{n\sigma^2_i} $$

The random variable $\Omega$ has a chi-square ($\chi^2$) distribution with $\frac{N}{n}$ degrees of freedom [32]. $\Omega$ thus has a mean of $\frac{N}{n}$ and a variance of $2\frac{N}{n}$. An estimate for $\sigma^2_i$ is obtained by multiplying $\Omega$ by the quantity $\frac{n}{N}$ to get

$$ \Lambda_i = \sigma^2_i \frac{n}{N} \Omega = \frac{1}{N} \sum_{k=1}^{N/n} (M_k - n)^2. $$

The mean of estimate $\Lambda_i$ is $\sigma^2_i$ and the variance is $2\frac{n}{N} (\sigma^2_i)^2$. If a decoding error does occur, only a relatively small number of trellis branches are likely to be affected and the estimate will remain good.

The estimate of the channel noise variance formed in this manner is extremely good for moderately large packets ($N/n \gg 10$). The central limit theorem indicates that the distribution of the estimate $\Lambda_i$ is essentially Gaussian. In addition, for channels with $\text{SNR}_{\text{dB}} > -3$ dB (most practical channels), the channel noise variance $\sigma^2_i$ is always less
than one. Therefore

\[ (\sigma_i^2)^2 < \sigma_i^2, \]

and hence

\[ \text{Var}(\Lambda_i) = \frac{2n}{N} (\sigma_i^2)^2 \ll \sigma_i^2. \]

The variance of the estimate is much less than the variance of the received symbols within the packet.

The quality of the estimate of the variance is quantified using two approximations developed by Papoulis [33, page 154]. Papoulis showed that if the function of two random variables \( g(x, y) \) is sufficiently smooth near the point \((\eta_x, \eta_y)\), where \( \eta_x \) and \( \eta_y \) are the means of \( x \) and \( y \) respectively, then the mean \( \eta_g \) and variance \( \sigma_g^2 \) of \( g(x, y) \) can be approximated using the following expressions:

\[
\eta_g \approx g + \frac{1}{2} \left( \frac{\partial^2 g}{\partial x^2} \sigma_x^2 + 2 \frac{\partial^2 g}{\partial x \partial y} r \sigma_x \sigma_y + \frac{\partial^2 g}{\partial y^2} \sigma_y^2 \right) \quad (46)
\]

\[
\sigma_g^2 \approx \left( \frac{\partial g}{\partial x} \right)^2 \sigma_x^2 + 2 \left( \frac{\partial g}{\partial x} \right) \left( \frac{\partial g}{\partial y} \right) r \sigma_x \sigma_y + \left( \frac{\partial g}{\partial y} \right)^2 \sigma_y^2, \quad (47)
\]

where \( \sigma_x^2 \) and \( \sigma_y^2 \) are the variances of \( x \) and \( y \) respectively, \( r \) is the correlation coefficient between \( x \) and \( y \), and the function \( g(x, y) \) and its derivatives are evaluated at \( x = \eta_x \) and \( y = \eta_y \).

When the combined packet is formed using perfect channel information and weighted diversity combining the following expression is used:

\[
z_j = \left[ \sum_{i=1}^{L} \frac{1}{\sigma_i^2} \right]^{-1} \sum_{i=1}^{L} \frac{1}{\sigma_i^2} y_{ij}. \quad (48)
\]

The summands \( \frac{1}{\sigma_i^2} y_{ij} \) are Gaussian random variables with mean \( x_i/\sigma_i^2 \) and variance \( 1/\sigma_i^2 \). When the estimate \( \Lambda_i \) of the channel variance is used instead of the actual variance the combining formula becomes

\[
z_i = \left[ \sum_{i=1}^{L} \frac{1}{\Lambda_i} \right]^{-1} \sum_{i=1}^{L} \frac{1}{\Lambda_i} y_{ij}. \quad (49)
\]

The mean of \( z_j \) is still \( x_j \), but the relative weighting of the packets is affected slightly. Note that the summands \( \frac{1}{\Lambda_i} y_{ij} \) are now functions \( g(y_{ij}, \Lambda_i) \) of two random variables. The
estimate \( A_i \) is actually a function of \( y_{ij} \), but for a moderately long packet the correlation coefficient \( r \) is approximately zero. Using Papoulis' formulae the mean and variance of the summands become

\[
\eta_s \approx \frac{x_i}{\sigma_i^2} \left( 1 + \frac{2n}{N} \right), \quad (50)
\]
\[
\sigma_s^2 \approx \frac{1}{N \sigma_i^2} \left( 1 + \frac{2n}{N \sigma_i^2} \right). \quad (51)
\]

For example, for a rate 1/2 code \((n = 2)\) with packet length \( N = 1000 \), the error \( \epsilon = \frac{n}{N \sigma_i^2} \) in the mean and variance of the summands is less than 0.02 (2%) if \( \sigma_i^2 > 0.1 \) \((\text{SNR}_{dB} < 7 \text{ dB})\). A channel with \( \text{SNR}_{dB} > 7 \text{ dB} \) allows for reliable decoding without combining and thus does not present a problem. The estimate of the channel variance is thus very good and is considered nearly optimal for low SNR's where combining generates the most benefit.

To obtain an estimate of the channel condition, an attempt must be made to decode the current copy of the packet before combining it with the previously received copies of the packet. If the current copy of the packet is decoded reliably then the decoded packet is accepted and no combining is used. The performance of the weighted diversity combiner using side information thus differs slightly from the ideal weighted diversity combining scheme.

The expected number of transmissions of a packet for the weighted diversity combining system using decoder side information is

\[
Tr = 1 + P(R_1') + P(R_1', R_2') + \cdots + P(R_1', R_2', \ldots, R_n') + \cdots, \quad (52)
\]

where \( R_n' \) is the event that a retransmission is requested by the decoder when decoding either the \( n^{th} \) copy of the packet or the \( n \) combined copies of the packet. Therefore \( P(R_n') = P(R_1, R_n) \) and \( P(R_1') = P(R_1) \). It is easily shown that

\[
P(R_1', R_2', \ldots, R_n') \geq P(R_1')P(R_2')\cdots P(R_n'). \quad (53)
\]

By noting that

\[
P(R_1') = P(R_1, R_1') = P(R_1)P(R_1|R_1) \geq P(R_1)P(R_1), \quad (54)
\]
it is seen that

\[ P(R'_1, R'_2, \ldots, R'_n) \geq [P(R_1)]^{n-1} \prod_{i=1}^{n} P(R_i). \]  

(55)

A lower bound on \( Tr \) is thus provided by

\[ Tr \geq 1 + \sum_{n=1}^{\infty} [P(R_1)]^{n-1} \prod_{i=1}^{n} P(R_i). \]  

(56)

An upper bound on \( P(R'_1, R'_2, \ldots, R'_n) \) is found in the following manner:

\[
P(R'_1, R'_2, \ldots, R'_n) = P(R'_n) P(R'_1, R'_2, \ldots, R'_{n-1}|R'_n)
\leq P(R'_n)
= P(R_1, R_n)
= P(R_n) P(R_1|R_n)
\leq P(R_n).
\]

(57)

Thus the expected number of retransmissions is upper bounded by

\[ Tr \leq 1 + \sum_{n=1}^{\infty} P(R_n). \]  

(58)

The probability of decoded bit error is also slightly affected by the implementation of the system using decoder side information. After each copy of a packet is received there is a possibility that the packet is decoded using only the most recently received copy of the packet. The probability of decoded bit error given the packet is decoded after the \( n^{th} \) copy has been received is

\[
P(B'_n) = \frac{P(B_1)P(A_1) + P(B_n)P(A_n)}{P(A_1) + P(A_n)}
= \frac{P(B_1)[1 - P(R_1)] + P(B_n)[1 - P(R_n)]}{2 - P(R_1) - P(R_n)},
\]

(59)

where \( P(B_n) \) is given by equation (13). The bounds on total probability of decoded bit error for the system using decoder side information are [24]

\[
P_B \leq P(B_1) + \sum_{i=1}^{\infty} \left( [P(R_1)]^{i-1} \prod_{j=1}^{i} P(R_j) \right) [P(B_{i+1}) - P(B_i)]
\]

(60)
and

\[ P_B \geq P(B_1) + \sum_{i=1}^{\infty} P(R_i)[P(B_{i+1}) - P(B_i)]. \] (61)

The weighted combining system using decoder side information has a lower expected number of retransmissions (higher throughput) and greater probability of bit error than the weighted combining system using perfect channel information. This difference is due to the initial decoding attempt on each received packet copy in the former system. The impact on throughput is shown in the next section to be very small.

5 Analytical and Simulation Results

To determine the tightness of the bounds derived in the previous section, simulation results have been compiled for the two combining systems. To simplify the comparison of the results, the same (2,1,3) convolutional code (rate 1/2, memory length 3) is used in each of the examples. The packet length is fixed at 1,000 coded bits for all systems considered.

The generating function for the (2,1,3) code is

\[ T(X,Y,Z) = \frac{X^6Y^2Z^5 + X^7YZ^4 - X^8Y^2Z^5}{1 - XY(Z + Z^2) - X^2Y^2(Z^4 - Z^3) - X^3YZ^3 - X^4Y^2(Z^3 - Z^4)}. \] (62)

The parameters of the code needed to calculate the bounds derived in the previous two sections are:

- \( d_{\text{free}} = 6, \)
- \( a(i : i = d_{\text{free}}, d_{\text{free}} + 1, d_{\text{free}} + 2, \ldots) = \{1, 3, 5, 11, 25, 55, 121, 267, 589, 1299, 2865, 6319, 13937, 30739, 67797, 145649, 298455, 572683, 942999, 1216286, \ldots\}, \)
- \( c(i : i = d_{\text{free}}, d_{\text{free}} + 1, d_{\text{free}} + 2, \ldots) = \{2, 7, 18, 49, 130, 333, 836, 2069, 5060, 12255, 29444, 70267, 166726, 393635, 925334, 2100931, 4502972, 8956629, 15139128, 19818994, \ldots\}. \)

The code above was selected because it offered a moderate amount of error correction/detection capacity while still being amenable to a thorough analysis without resort
to simplifying assumptions. Several other codes were simulated as well as the (2,1,3) code, and all showed the same basic tendencies discussed in the following pages.

The results provided in the following discussion result from the simulated transmission of 10,000 packets of 1,000 bits each for the stationary or slowly varying channel cases, and 1,000 packets of 1,000 bits each for the moderately-varying channel case. The software implementation of the Viterbi decoder completed the decoding of an entire packet before releasing decoded bits, so there were no truncation effects. Quantization effects were also minimized through the use of eight bit vectors to represent the soft-decision values of the received bits.

5.1 Averaged Diversity Combining Over Stationary and Slowly-Varying Channels

It is assumed that averaged diversity combining is incorporated into a soft-decision Viterbi decoder implementing the Yamamoto-Itoh algorithm. The following computed and simulated results are for the AWGN channel case.

Lower bounds for the throughput of the averaged diversity combining system are plotted in Figure 1. The simulation results and the lower bounds for the throughput of the Yamamoto-Itoh algorithm without combining are also plotted for comparison. Note that the systems with combining provide significant throughput far below the signal-to-noise ratios at which the throughput for the systems without combining becomes negligible.

Figure 2 shows the upper bounds on the decoded BER for systems using the Yamamoto-Itoh hybrid-ARQ algorithm with and without averaged diversity combining. The bounds on the decoded BER for systems using the combining algorithm are plotted only for SNR’s where a good approximation of the value of the probability of retransmission is available. From the throughput results for the Yamamoto-Itoh algorithm without combining in Figure 1, it is found that the upper bounds on the probability of retransmission approximate the actual probability of retransmission for SNR’s greater than or equal to 0.5 dB, 1.5 dB, and 2.5 dB for thresholds of 1, 3, and 5 respectively. Note that the bounds on the decoded
Figure 1: Throughput for Averaged Diversity Combining over a Stationary AWGN Channel
BER for the averaged diversity combining system are always less than or equal to the decoded BER bounds of the system without combining. The increased throughput performance resulting from the averaged diversity combining does not increase the decoded BER of the system.

As expected, the addition of averaged diversity combining to a soft decision Viterbi decoder with type-I hybrid-ARQ capability greatly improves the performance of the system. For example, assuming a threshold \( u \) of 5, the system employing averaged diversity combing maintains a throughput greater than 0.2 for SNR's greater than or equal to 1 dB. The throughput of the system without code combining has negligible throughput over channels with SNR's less than 2 dB. The upper bounds on the decoded BER in Figure 2 also indicate that the decoded BER for systems employing code combining is much lower than systems without code combining for low channel SNR's.

5.2 Moderately-Varying Channels

Four time-varying channel models were used to establish the performance characteristics of the combining algorithms defined in Section 4. Each of the channel models are described in terms of the pdf of the channel's SNR in dB. The pdf of the first channel is uniformly distributed between 1 and 7 dB. The pdf of the second channel is Gaussian with a mean of 3 dB and a variance of 2 dB. An offset Rayleigh distribution is used for the third channel. The offset Rayleigh distribution is a simple Rayleigh distribution with \( \sigma^2 = 2 \) dB that is offset such that the minimum is 1 dB (instead of 0 dB). The pdf of the fourth channel is bimodal with equal impulses at 0 and 4 dB. Sketches of each of these channel models are provided in Figure 3. The performance bound computations for the systems using averaged diversity combining and weighted diversity combining require that the pdf of the channel models be in terms of the channel noise variance and the linear SNR of the channel, respectively. The pdf's for the channel's linear SNR and noise variance are derived and used in the computation of the performance bounds.

The four channel models were selected so as to be tractable analytically, but highly
Figure 2: Upper Bounds on Decoded BER for Averaged Diversity Combining over a Stationary AWGN Channel
Figure 3: Distributions of Non-Stationary Channels
disparate in character. For example, the Gaussian distributed channel is compact and has a very definite mean whose approximate value is frequently assumed by the channel. The bimodal channel, on the other hand, is highly spread out and has a mean that is never assumed by the channel. This distinction has an interesting impact on some of the results that follow.

The performance bounds are computed in the following manner. The pdf of the effective noise variance or the linear SNR for $L$ combined packets is calculated using the inverse Fourier transform of the $L^{th}$ power of the Fourier transform of the single packet pdf. The probability of packet acceptance given $L$ combined packets is then found by numerical integration of equation (10). The probability of decoded bit error for $L$ combined packets is found by numerical integration on equation (13). The number of points used in each numerical integration and the maximum frequency of the characteristic function used in the calculation of the effective PDFs were selected through trial and error to give a 3 decimal place accuracy in the expected throughput and a 2 digit accuracy in the decoded BER.

Simulations were used to verify the throughput bounds calculated for the Yamamoto and Itoh system without combining, with averaged diversity combining, with weighted diversity combining with perfect side information, and with weighted diversity combining using decoder side information. Due to the extremely good reliability performance provided by the hybrid-ARQ systems (with and without packet combining), none of the simulations had a sufficient number of decoded bit errors to obtain a value for the decoded BER.

Tables 1 – 8 list the results of the computation of the lower bounds on throughput, simulated throughput, and the computation of upper bounds on decoded BER for each of the channel models and coding systems. The following abbreviations are used in the tables to indicate the type of error control system used:

- $\text{YI}$ – Yamamoto-Itoh type-I hybrid-ARQ algorithm without any combining,
- $\text{AD}$ – $\text{YI}$ with averaged diversity combining,
• WD – YI with weighted diversity combining with perfect side information,

• IWD – YI with imperfect weighted diversity combining using decoder side information.

In Tables 1, 3, 5 and 7 the simulated throughput is listed in parenthesis under the calculated lower bound. For the cases where the lower bound on expected throughput is less than 0.100, the simulations were prohibitively time consuming and were therefore not completed. Sufficient numbers of simulations were, however, completed to verify the bounds derived in the previous sections.

Several interesting results become apparent when viewing the throughput results in Tables 1, 3, 5 and 7. First, averaged diversity combining, as in the case of the stationary and slowly-varying channels, provides large increases in throughput when used over moderately-varying channels. For instance, the throughput for the Yamamoto-Itoh algorithm with threshold $u = 5$ over a channel with Gaussian distributed SNR results in a simulated throughput of 0.180 (see Table 3). The addition of averaged diversity combining improves the system's simulated throughput performance to 0.294. This is an improvement of 68 percent over the system without combining. Therefore the simple averaged diversity combining protocol provides a considerable increase in throughput over hybrid-ARQ systems with relatively low throughput without relying on any channel side information.

Another important result is the relative performance between the averaged diversity combining systems and the systems using weighted diversity combining with perfect side information. Chase [13] determined that weighted diversity combining was optimum when the channel conditions were varying. The analyses and simulations performed here verify Chase's conclusions, but for many cases weighted diversity combining provides small or even negligible increases in throughput over averaged diversity combining. In fact in all the cases where the throughput for the systems using averaged diversity scheme is at least 0.250, the increase in throughput afforded by systems using weighted diversity combining is less than 2 percent.

The reason for the small improvement provided by weighted diversity combining can
be seen by considering the effective SNR of the packet formed by combining two copies of a packet transmitted over channels with different SNR's. Assume that one copy is transmitted over a channel with SNR = 0 dB and the second copy over a channel with 4 dB SNR. If the two copies are combined using averaged diversity combining, then the effective SNR of the combined packet is 4.56 dB. The effective SNR of the combination using weighted diversity combining is 5.46 dB. If for example the threshold \( u \) is set at 5, then the probability of accepting the packet combined using averaged diversity combining is approximately 0.9 and the probability of accepting the packet combined using weighted diversity combining is approximately 0.98. Both have a very high probability of accepting the two packets combined and, in an adaptive rate system where the acceptance of the first copy is independent of the combining method used, the two combining systems have nearly identical throughput. In Table 7 it can be seen that for the case \( u = 5 \) the throughput of the weighted diversity combining system has only 1.5 percent higher throughput than the system using averaged diversity combining.

The systems using weighted diversity combining do have a significantly higher throughput than systems using averaged diversity combining when the throughput becomes relatively low. For example, in Table 7 the system with \( u = 7 \) using weighted diversity combining has a 10 percent higher throughput than the system using averaged diversity combining over a channel with bimodal SNR. For the case of a system with \( u = 7 \) over a channel with Rayleigh distributed SNR, the throughput of the weighted system is 6 percent higher than the averaged combining system. In each of these cases, the throughput for the averaged diversity combining systems is less than 0.250 and thus there is a significant probability that 3 or more copies of a packet will need to be transmitted before the packet is accepted. The advantage of weighted diversity combining over averaged diversity combining is greater when more copies of each packet are combined.

The throughput of the weighted diversity combining protocol using decoder side information performed nearly the same as the protocol using perfect side information. In all cases the throughput for the systems using decoder side information is within 2.5 percent
of the throughput for the systems using perfect side information (the worst case is seen in Table 7 for \( u = 5 \)). The use of estimates for the channel SNR or noise variance derived from decoder side information thus does not appear to appreciably degrade the performance of the weighted combining systems.

The upper bounds on decoded BER for most of the cases listed in Tables 2, 4, 6 and 8 reflect the improvement in decoded BER encountered in ARQ protocols when the number of retransmission requests is reduced while all other parameters remain fixed. The only unexpected results are for the bimodal channel in Table 8. For the systems using averaged diversity combining with \( u = 4, 5 \) or 6, the upper bound on the decoded BER was higher than the corresponding systems without code combining. At first the results seem incongruous, but upon further reflection they can be explained. For each of these cases the probability of accepting the first copy of the packet when transmitted over a channel with 0 dB SNR is very small. But, if 2 copies transmitted over a channel with 0 dB SNR are combined, then there is a significant probability that the combined packet is accepted. The combination of two copies of a packet transmitted over a channel with 0 dB SNR has an effective SNR of 3 dB. Therefore the system using combining has a significant probability of accepting a combined packet with 3 dB effective SNR while the system without combining almost always waits for a packet transmitted over a channel with 4 dB SNR. Accepting a combined packet with an effective SNR of 3 dB results in a higher decoded BER than accepting a packet transmitted over a channel with 4 dB SNR. It is thus possible to have a higher decoded BER for a combining system than for a system without combining. ADC systems are the only examples in Table 8 for which combining systems are outperformed by non-combining systems, but it should be noted that the same situation can arise with weighted diversity combining systems. Note also that these are only bounds and that the actual decoded BER's may not differ as dramatically as indicated by the bounds.
Table 1: Throughput Lower Bounds and Simulation Results for Uniformly Distributed Channel SNR

<table>
<thead>
<tr>
<th>$u$</th>
<th>$YI$</th>
<th>$AD$</th>
<th>$WD$</th>
<th>$IWD$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.493 (0.494)</td>
<td>0.494 (0.494)</td>
<td>0.494 (0.494)</td>
<td>0.494 (0.494)</td>
</tr>
<tr>
<td>2</td>
<td>0.477 (0.477)</td>
<td>0.478 (0.478)</td>
<td>0.478 (0.478)</td>
<td>0.478 (0.478)</td>
</tr>
<tr>
<td>3</td>
<td>0.435 (0.436)</td>
<td>0.442 (0.441)</td>
<td>0.442 (0.441)</td>
<td>0.442 (0.441)</td>
</tr>
<tr>
<td>4</td>
<td>0.365 (0.367)</td>
<td>0.393 (0.394)</td>
<td>0.393 (0.393)</td>
<td>0.393 (0.390)</td>
</tr>
<tr>
<td>5</td>
<td>0.278 (0.282)</td>
<td>0.342 (0.342)</td>
<td>0.343 (0.349)</td>
<td>0.343 (0.349)</td>
</tr>
<tr>
<td>6</td>
<td>0.174 (0.186)</td>
<td>0.283 (0.291)</td>
<td>0.288 (0.291)</td>
<td>0.288 (0.296)</td>
</tr>
<tr>
<td>7</td>
<td>0.023 (*)</td>
<td>0.212 (0.218)</td>
<td>0.223 (0.230)</td>
<td>0.223 (0.223)</td>
</tr>
</tbody>
</table>

* - not simulated.

Table 2: Upper Bounds on Decoded BER for Uniformly Distributed Channel SNR

<table>
<thead>
<tr>
<th>$u$</th>
<th>$YI$</th>
<th>$AD$</th>
<th>$WD$</th>
<th>$IWD$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$3.6 \times 10^{-6}$</td>
<td>$3.5 \times 10^{-6}$</td>
<td>$3.5 \times 10^{-6}$</td>
<td>$3.6 \times 10^{-6}$</td>
</tr>
<tr>
<td>2</td>
<td>$5.9 \times 10^{-7}$</td>
<td>$5.6 \times 10^{-7}$</td>
<td>$5.6 \times 10^{-7}$</td>
<td>$5.8 \times 10^{-7}$</td>
</tr>
<tr>
<td>3</td>
<td>$5.4 \times 10^{-8}$</td>
<td>$4.7 \times 10^{-8}$</td>
<td>$4.7 \times 10^{-7}$</td>
<td>$5.3 \times 10^{-8}$</td>
</tr>
<tr>
<td>4</td>
<td>$1.3 \times 10^{-9}$</td>
<td>$9.4 \times 10^{-10}$</td>
<td>$9.4 \times 10^{-10}$</td>
<td>$1.2 \times 10^{-9}$</td>
</tr>
<tr>
<td>5</td>
<td>$3.8 \times 10^{-12}$</td>
<td>$2.1 \times 10^{-12}$</td>
<td>$2.1 \times 10^{-12}$</td>
<td>$3.3 \times 10^{-12}$</td>
</tr>
<tr>
<td>6</td>
<td>$9.9 \times 10^{-16}$</td>
<td>$3.5 \times 10^{-16}$</td>
<td>$3.5 \times 10^{-16}$</td>
<td>$6.9 \times 10^{-16}$</td>
</tr>
<tr>
<td>7</td>
<td>$7.4 \times 10^{-21}$</td>
<td>$9.4 \times 10^{-22}$</td>
<td>$9.4 \times 10^{-22}$</td>
<td>$2.6 \times 10^{-21}$</td>
</tr>
</tbody>
</table>
Table 3: Throughput Lower Bounds and Simulation Results for Gaussian Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Throughput Lower Bounds (simulated)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>0.476 (0.483)</td>
</tr>
<tr>
<td>2</td>
<td>0.448 (0.454)</td>
</tr>
<tr>
<td>3</td>
<td>0.392 (0.400)</td>
</tr>
<tr>
<td>4</td>
<td>0.299 (0.313)</td>
</tr>
<tr>
<td>5</td>
<td>0.175 (0.180)</td>
</tr>
<tr>
<td>6</td>
<td>0.064 (*)</td>
</tr>
<tr>
<td>7</td>
<td>0.010 (*)</td>
</tr>
</tbody>
</table>

* - not simulated.

Table 4: Upper Bounds on Decoded BER for Gaussian Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Decoded BER Upper Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>$1.8 \times 10^{-5}$</td>
</tr>
<tr>
<td>2</td>
<td>$1.7 \times 10^{-6}$</td>
</tr>
<tr>
<td>3</td>
<td>$8.8 \times 10^{-8}$</td>
</tr>
<tr>
<td>4</td>
<td>$1.7 \times 10^{-9}$</td>
</tr>
<tr>
<td>5</td>
<td>$8.2 \times 10^{-12}$</td>
</tr>
<tr>
<td>6</td>
<td>$4.4 \times 10^{-15}$</td>
</tr>
<tr>
<td>7</td>
<td>$6.7 \times 10^{-20}$</td>
</tr>
</tbody>
</table>
Table 5: Throughput Lower Bounds and Simulation Results for Rayleigh Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Throughput Lower Bounds (simulated)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>0.493 (0.496)</td>
</tr>
<tr>
<td>2</td>
<td>0.471 (0.484)</td>
</tr>
<tr>
<td>3</td>
<td>0.405 (0.456)</td>
</tr>
<tr>
<td>4</td>
<td>0.272 (0.375)</td>
</tr>
<tr>
<td>5</td>
<td>0.119 (0.246)</td>
</tr>
<tr>
<td>6</td>
<td>0.027 (*)</td>
</tr>
<tr>
<td>7</td>
<td>0.002 (*)</td>
</tr>
</tbody>
</table>

* - not simulated.

Table 6: Upper Bounds on Decoded BER for Rayleigh Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Decoded BER Upper Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>$3.3 \times 10^{-6}$</td>
</tr>
<tr>
<td>2</td>
<td>$5.3 \times 10^{-7}$</td>
</tr>
<tr>
<td>3</td>
<td>$5.8 \times 10^{-8}$</td>
</tr>
<tr>
<td>4</td>
<td>$2.5 \times 10^{-9}$</td>
</tr>
<tr>
<td>5</td>
<td>$2.0 \times 10^{-11}$</td>
</tr>
<tr>
<td>6</td>
<td>$1.5 \times 10^{-14}$</td>
</tr>
<tr>
<td>7</td>
<td>$3.2 \times 10^{-19}$</td>
</tr>
</tbody>
</table>
Table 7: Throughput Lower Bounds and Simulation Results for Bimodal Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Throughput Lower Bounds (simulated)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>0.324 (0.368)</td>
</tr>
<tr>
<td>2</td>
<td>0.259 (0.290)</td>
</tr>
<tr>
<td>3</td>
<td>0.248 (0.257)</td>
</tr>
<tr>
<td>4</td>
<td>0.233 (0.237)</td>
</tr>
<tr>
<td>5</td>
<td>0.170 (0.168)</td>
</tr>
<tr>
<td>6</td>
<td>0.041 (*)</td>
</tr>
<tr>
<td>7</td>
<td>0.000 (*)</td>
</tr>
</tbody>
</table>

* - not simulated.

Table 8: Upper Bounds on Decoded BER for Bimodal Distributed Channel SNR

<table>
<thead>
<tr>
<th>u</th>
<th>Decoded BER Upper Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>YI</td>
</tr>
<tr>
<td>1</td>
<td>2.9 x 10^{-4}</td>
</tr>
<tr>
<td>2</td>
<td>1.5 x 10^{-5}</td>
</tr>
<tr>
<td>3</td>
<td>2.6 x 10^{-8}</td>
</tr>
<tr>
<td>4</td>
<td>3.1 x 10^{-13}</td>
</tr>
<tr>
<td>5</td>
<td>8.2 x 10^{-15}</td>
</tr>
<tr>
<td>6</td>
<td>2.1 x 10^{-16}</td>
</tr>
<tr>
<td>7</td>
<td>4.3 x 10^{-18}</td>
</tr>
</tbody>
</table>
6 Conclusions

A type-I hybrid-ARQ protocol can provide better reliability performance than an FEC system using the same code. The cost of the increased reliability is a reduction in throughput caused by retransmissions as the communication channel degrades. Packet combining recovers a considerable portion of the lost throughput by varying the effective code rate of the error control system. As each successive copy of a packet is received it is combined with all previously received copies to form a new code word which, in most cases, has a greater effective SNR (equivalent to a lower effective channel BER) than any of the received copies by themselves.

The combining protocols described here are designed such that only a single decoder is needed to decode any received packet or any combination of received copies of the packet. The implementation of any of these protocols requires only minor modifications of the transmitting and receiving systems. The majority logic combining system extends the operating range of hard decision Viterbi decoders to BER's greater than 0.1, while the upper bounds on the decoded BER for the majority logic combining system remain lower than the bounds for the system using hybrid-ARQ alone.

Averaged diversity combining increases the range of stationary and slowly-varying channel SNRs over which significant throughput can be achieved by type-I hybrid-ARQ protocols using soft decision Viterbi decoders. At low channel SNR's the averaged diversity combiner provides significantly better throughput performance than simple hybrid-ARQ systems while providing reliability performance that is at least as good.

Over moderately-varying channels the simple averaged diversity combiner provides a great improvement in throughput without the added complexity of channel state estimation. In several simulated examples the averaged diversity combining improved the throughput by over 50 percent when compared to type-I hybrid-ARQ alone.

Over moderately-varying channels where there is a significant probability that 3 or more copies of a packet will be required before reliable decoding is possible, weighted diversity combining provides improvement in throughput over averaged diversity combin-
ing. Optimal weighting factors are derived from ideal channel information. These weights increase the probability of acceptance of the combined copies of a packet over that of a packet formed through averaged diversity combining.

Weighting factors for weighted diversity combining can be derived from the path metrics generated by the Viterbi decoder. The simulation results show that the weighted combining systems using decoder side information have throughputs very near that of systems which have access to perfect channel state information. A weighted diversity combining can thus be implemented in a convolutionally encoded system using a Viterbi decoder with type-I hybrid-ARQ without the additional complexity of channel monitoring hardware.

References


Type-II Hybrid-ARQ Protocols Using Punctured MDS Codes

Stephen B. Wicker†
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Michael Bartz
Department of Electrical Engineering
Memphis State University
Memphis, Tennessee 38152

Abstract

MDS codes possess several properties that make them an ideal choice for type-II hybrid-ARQ protocols. These properties include "strong separability", "strong invertibility", and excellent reliability performance when used for simultaneous error detection and correction. In this paper these properties are shown to lead to a natural definition of an MDS type-II hybrid-ARQ protocol. An \((n,k)\) MDS code is decomposed into a pair of \((n/2,k)\) punctured MDS codes. The original code and the two derivative codes are used individually in type-I hybrid-ARQ protocols. These three type-I protocols combine to form a single type-II protocol. The performance of this system is analyzed in detail, with particular attention paid to the definition of an effective channel model for code words that are known to have caused the generation of retransmission requests.

1 Introduction

Maximum distance separable (MDS) codes provide excellent reliability performance when used for error detection or combined error detection and error correction [1]. They are thus natural candidates for use in hybrid-ARQ protocols [2], in which error detection and

†This work was supported by the National Science Foundation under Grant Number NCR-9009877.
correction are combined in a receiver that detects unreliable decoded code words and requests their retransmission. It has been noted, however, that the combined error detection and correction capabilities of MDS codes can become a liability when the communication channel is nonstationary [3]. Consider a type-I hybrid-ARQ protocol that has been designed for a fixed channel noise level. In a type-I protocol each transmitted code word is encoded for both error detection and error correction. The error correction capacity is used to correct frequently occurring error patterns, while the detection capacity is used to detect the less frequently occurring patterns, which cause the generation of retransmission requests. In a type-I protocol the transmitter responds to retransmission requests by sending another copy of the transmitted code word. This error control scheme performs quite well on channels that are essentially stationary except for infrequent bursts of additional noise. However, if the channel noise level deviates from the design level for a significant period of time, the performance of the protocol can be seriously degraded. As the channel noise level increases, the probability that each received word contains an uncorrectable error pattern also increases. The new error patterns are detected, and a flood of retransmission requests ensues that persists until the channel noise level returns to its original level. As the channel noise level decreases, the error correction capacity is sufficient to correct all error patterns, rapidly driving the frequency of retransmission requests to zero. The redundancy reserved for error detection thus assumes the status of useless overhead. In either case the throughput performance of the type-I protocol is suboptimal. This problem is particularly acute when the type-I protocol is based on codes whose error correction and detection performance curves have strongly negative slopes as a function of channel noise (e.g. MDS codes).

Code combining offers a solution to this problem. The code combining receiver concatenates received code words until their combined code rate is sufficient to reliably recover the transmitted information [4]. As the channel noise level varies, the receiver varies the effective code rate of the error control system, reducing the throughput degradation observed with a fixed-rate system. The simplest code combining system is the type-II hybrid-ARQ protocol, a truncated form that limits combining operations to a maximum of two received code words [5],[6],[7]. In a type-II protocol the transmitter responds to an initial retransmission request by transmitting a code word containing parity bits for the first code word. The original message is obtained through decoding operations on the first
or second code words alone, or through a combined decoding operation on the composite code word created through the concatenation of the two received code words.

MDS codes possess a number of properties that make them well suited for use in type-II protocols. Mandelbaum [8], [9] has noted that Reed-Solomon codes (members of the MDS family) can be punctured to provide a primary code word and one or more secondary blocks that provide incremental redundancy as needed. This scheme is optimal in the sense that the incremental redundancy increases the minimum distance of the composite received word by the greatest possible amount per additional symbol. This paper modifies and extends Mandelbaum's work by defining a type-II hybrid-ARQ protocol based on the general class of punctured MDS codes.

Pursley and Sandberg have proposed the use of Reed-Solomon codes in an incremental redundancy system for meteor-burst channels [10], [11]. In this paper we consider their version of the RS type-II system as well as a modified version with fewer decoding operations. The analytical framework presented here can be used to accurately predict the performance of both systems.

In Section 2 an analysis of the reliability and throughput performance of a generic type-II hybrid-ARQ protocol is provided. A general review of the relevant properties of MDS codes follows. It is then shown that the various properties of MDS codes can be used to construct a type-II protocol from a series of type-I protocols based on punctured MDS codes. The performance parameters of the individual type-I protocols provide the necessary data for the complete characterization of the type-II system using the general expressions derived in the earlier section. Several examples are provided to indicate the excellent throughput and reliability performance offered by the MDS type-II hybrid-ARQ protocol.

2 Performance Model for the General Type-II Hybrid-ARQ Protocol

In a type-II hybrid-ARQ protocol, a code word is encoded using two codes, \( C_1 \) and \( C_2 \), to create a pair of code words \( c_1 \) and \( c_2 \). These codes have corresponding decoding operations \( D_1 \) and \( D_2 \) which can recover the original code word from noise corrupted versions of \( c_1 \) and \( c_2 \) respectively. \( c_1 \) comprises the initial transmission while \( c_2 \) is set aside. Upon
Figure 1: Flowchart for a type-II hybrid-ARQ protocol
receiving \( c_1 \), the receiver attempts decoding operation \( D_1 \) to recover the transmitted data. If the attempt is successful (i.e. no retransmission request is generated), the receiver sends an acknowledgement (ACK) to the transmitter; otherwise, a negative acknowledgement (NACK) is sent. The transmitter responds to the NACK by sending \( c_2 \). The receiver then attempts to decode \( c_2 \) by itself using decoding operation \( D_2 \). If successful, the receiver inverts the corrected version of \( c_2 \) to recover the desired information and sends an ACK to the receiver. If unsuccessful, the receiver combines \( c_1 \) and \( c_2 \) to create \( c_3 \), a code word in a lower rate code \( C_3 \). If the third decoding operation \( D_3 \) is successful, the data is recovered and an ACK is sent to the receiver; otherwise, a NACK is sent and the entire process is repeated. After the first pair of transmissions, the combined decoding operation \( D_3 \) is always available after the receipt of subsequent copies of either \( c_1 \) or \( c_2 \). This decoding protocol is shown as a flowchart in Figure 1. Note that this protocol is a slight generalization of the type-II protocol originally presented by Lin and Yu [5].

Reliability and throughput generating functions for this protocol are obtained using signal flow graph techniques [12]. A generic graph that describes this protocol is depicted in Figure 2. The nodes of the graph consist of the initial transmission \( IT \), code word acceptance \( CA \), and the decoding operations \( D_1 \), \( D_2 \), and \( D_3 \). The branches indicate the directions by which the code word transmission and decoding processes proceed. For reliability analysis the branches are labeled with the probability that the associated
Table 1: Graph labels for the derivation of throughput and reliability generating functions

<table>
<thead>
<tr>
<th>Branch label</th>
<th>Throughput label</th>
<th>Reliability label</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>$T^{n_1}$</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>$1 - p_R^{(1)}$</td>
<td>$p_E^{(1)}$</td>
</tr>
<tr>
<td>c</td>
<td>$p_R^{(1)} \cdot T^{n_2}$</td>
<td>$p_R^{(1)}$</td>
</tr>
<tr>
<td>d</td>
<td>$1 - p_R^{(2)}$</td>
<td>$p_E^{(2)}$</td>
</tr>
<tr>
<td>e</td>
<td>$p_R^{(2)}$</td>
<td>$p_R^{(1)}$</td>
</tr>
<tr>
<td>f</td>
<td>$1 - p_R^{(3)}$</td>
<td>$p_E^{(3)}$</td>
</tr>
<tr>
<td>g</td>
<td>$p_R^{(3)} \cdot T^{n_1}$</td>
<td>$p_R^{(3)}$</td>
</tr>
<tr>
<td>h</td>
<td>$1 - p_R^{(1)}$</td>
<td>$p_E^{(1)}$</td>
</tr>
<tr>
<td>i</td>
<td>$p_R^{(2)}$</td>
<td>$p_R^{(1)}$</td>
</tr>
<tr>
<td>j</td>
<td>$1 - p_R^{(3)}$</td>
<td>$p_E^{(3)}$</td>
</tr>
<tr>
<td>k</td>
<td>$p_R^{(3)} \cdot T^{n_2}$</td>
<td>$p_R^{(3)}$</td>
</tr>
</tbody>
</table>

event occurs, whereas for throughput calculations, the branch labels also help determine the number of transmitted symbols. The branch labels for determining reliability and throughput are found in Table 1. The generic graph yields the following transfer function:

$$IT = \left\{ ab + ac (d + ef + egh + egij) \left( \frac{1}{1 - e_{gik}} \right) \right\} CA$$ (1)

Substituting the appropriate branch values, one obtains the throughput and reliability generating functions for the type-II protocol.

For throughput calculation the branches are labeled with the probabilities of the generation of a retransmission request $p_R$, decoder error $p_E$, and code word acceptance, $1 - p_R$, as appropriate. The superscripts for the various probabilities reference the probabilities to a specific decoding operation. The variable $T$ is used to indicate the transmission of a code word, while its superscript denotes the number of code word symbols contained in the code word (either $n_1$ or $n_2$ for code $C_1$ or $C_2$ respectively). For the calculation of the throughput, a selective repeat protocol is assumed. The throughput generating function is as follows:
\[ G(T) = T^{n_1} p_C^{(1)} + T^{n_1+n_2} p_R p_C^{(1)} + p_C^{(2)} + T^{n_1} p_R^{(2)} p_R^{(3)} p_C^{(1)} + p_C^{(3)} \left( p_R^{(2)} + T^{n_1} p_R^{(2)} p_R^{(3)} p_C^{(1)} \right) \left( \frac{1}{1 - T^{n_1+n_2} p_R^{(1)} p_R^{(2)} p_R^{(3)}} \right) \] (2)

Once the throughput generating function has been obtained, the throughput \( \eta \) of the protocol can be computed. The throughput \( \eta \) is defined here as the ratio of the number of information symbols transmitted (\( k \), the dimension of code \( C_1 \)) to the average number of symbols transmitted before the code word is correctly accepted. By taking the partial derivative of Equation (2) with respect to \( T \) and setting \( T \) equal to unity, the probability of each of the distinct paths through the graph in Figure 2 is weighted by the total number of symbols transmitted along that path. The following expression results.

\[ \eta = k \left( \frac{\partial}{\partial T} G(T) \right)_{|T=1}^{-1} = k \left\{ \frac{1 - p_R^{(1)} p_R^{(2)} p_R^{(3)} p_R^{(3)} p_R^{(3)}}{n_1 + n_2 p_R^{(1)} + n_1 p_R^{(1)} p_R^{(2)} p_R^{(3)} - n_1 p_R^{(1)} p_R^{(2)} p_R^{(3)} p_R^{(3)}} \right\} \] (3)

The reliability generating function provides the following expression for the probability that an accepted, decoded code word contains one or more symbol errors.

\[ P(E) = \left\{ p_E^{(1)} + p_R^{(1)} p_E^{(2)} + p_R^{(2)} p_E^{(3)} + p_R^{(2)} p_R^{(3)} p_E^{(1)} + p_R^{(1)} p_R^{(2)} p_R^{(3)} p_R^{(3)} \left( \frac{1}{1 - p_R^{(1)} p_R^{(2)} p_R^{(3)} p_R^{(3)}} \right) \right\} \] (4)

3 The Properties of MDS Codes

The use of MDS codes in type-II protocols is motivated by a series of properties that are unique to MDS codes. The most pertinent are listed here. The first property is frequently used as the definition for MDS codes, though it can be shown to be equivalent to a number of other definitions.
Property 1 The Singleton Bound states that given a linear code C with length $n$ and dimension $k$, the minimum distance $d_{\text{min}}$ must satisfy $d_{\text{min}} \leq (n - k + 1)$. A code C is MDS if and only if it satisfies the Singleton Bound with equality [13].

Property 1 shows that MDS codes are optimal in the sense that they provide "maximum distance" between code words. MDS codes were once called "optimal codes", but this proved to be confusing and was abandoned in later literature [14], [15]. The Singleton Bound can be used in conjunction with the BCH bound to show that Reed-Solomon codes are MDS [15].

When the "natural" length of a particular code is unsuitable for an application, the length can be changed by puncturing, extending, shortening, or lengthening the original code [14], [15], [16]. In this paper the technique of interest is puncturing. A code is punctured through the consistent deletion of parity coordinates from each code word in the code. Puncturing $j$ coordinates reduces an $(n, k, d)$ code to an $(n - j, k, d')$ code. In most cases the goal is to minimize the reduction in minimum distance through the judicious selection of the deleted coordinates. In the case of MDS codes, however, the minimum distance of the resulting code is solely a function of the number of coordinates punctured. Any combination of $j$ puncturing operations changes an $(n, k, n - k + 1)$ MDS code into an $(n - j, k, n - k - j + 1)$ MDS code.

Property 2 Punctured MDS codes are MDS.

This is easily proven by noting that the elimination of a coordinate in a code can reduce the code's minimum distance by at most one, while the Singleton Bound implies that the minimum distance of an MDS code must be reduced by at least one when the length is reduced by one. The result, of course, is a consistent reduction of the minimum distance by one with each successive puncturing operation.

Property 1 can be shown to imply a "separability" property which proves quite useful in the development of code combining schemes [13]. Unfortunately the term "separable" has enjoyed a variety of definitions in the literature that are not equivalent. In works related to MDS codes "separable" is taken to mean that a code can be partitioned (separated) into message symbols and parity symbols (i.e. a systematic representation of the code exists) [13], [15]. In this sense of the word, any linear code is separable, for a generator matrix G for an $(n, k)$ code must have at least one combination of $k$ linearly independent columns.
In works involving type-II hybrid-ARQ protocols, however, “separable” has been used to describe any code \{F(x)\} for which there is a punctured version \{f(x)\} that is “capable of detecting by itself a number of errors eventually correctable by \{F(x)\}” [17]. In this paper the former definition is adopted, for it is this sense of separability that leads to the construction of the desired MDS code combining protocol. An \((n, k)\) code shall be called strongly separable if any \(k\) code word coordinates can be used as the information symbols in a systematic representation.

**Property 3** MDS codes are strongly separable [13].

A code is said to be invertible if the parity-check symbols of the code word can be used by themselves to uniquely determine the information symbols through an inversion process [18]. An \((n, k)\) code shall be called strongly invertible if any \(k\) symbols from the code word can be used to recover the information symbols.

**Property 4** MDS codes are strongly invertible.

This property is proved in Appendix B.

The final property of interest is the MDS weight enumerator, which allows for an exact determination of the probabilities of undetected error and retransmission request.

**Property 5** The number of code words of weight \(j\) in an \((n, k, d_{\text{min}})\) \(2^m\)-ary MDS code is [14]

\[
A_j = \binom{n}{j} (2^m - 1) \sum_{i=0}^{j-d_{\text{min}}} (-1)^i \binom{j-i}{i} 2^m(j-i-d_{\text{min}}).
\]

### 4 Punctured MDS Codes in a Type-II Hybrid-ARQ Protocol

In the MDS type-II protocol, the codes \(C_1, C_2,\) and \(C_3\) are formed in a very natural manner. The first step is to select an \((n, k)\) MDS code with rate less than one-half for the combined code \(C_3\). Using decoding operation \(D_3\), this code should provide sufficient error correction capability for the reliable transmission of information under the worst channel
conditions expected. Figure 3 shows how code words from C₃ are punctured to form code words in C₁ and C₂. The first \( n/2 \) coordinates in a given C₃ code word \( c_{i,3} \) form the code word \( c_{i,1} \) in C₁, while the remaining \( n/2 \) coordinates form the code word \( c_{i,2} \) in C₂. Since codes C₁ and C₂ are punctured versions of the MDS code C₃, they are themselves MDS by Property 2. Property 4 guarantees that corrected versions of code words from any of the three codes can be used to recover the information symbols. Decoding operations \( D_1 \) and \( D_2 \) are designed so as to maximize throughput while maintaining a minimum allowable level of reliability under optimum channel conditions. The design of the individual decoding operations is developed in the following section.

5 A Retransmission Request Mechanism for MDS Codes

All three of the decoding operations used in a type-II protocol need a retransmission request mechanism to detect code words whose completed decoding will result in unreliable information symbols. In the MDS type-II scheme, the same retransmission request mechanism is used with all three decoding operations. All three are treated as type-I hybrid-ARQ protocols that combine to form a type-II protocol. In this section the design and analysis of the MDS type-I hybrid-ARQ protocol is discussed.

5.1 The MDS Type-I Hybrid-ARQ Protocol

In earlier papers a method was demonstrated for modifying FEC Reed-Solomon error control systems for use in type-I hybrid-ARQ protocols [2], [19, fading channels with erasure decoding]. These discussions are easily generalized for application to bounded distance decoders for MDS codes.

Given an MDS code with minimum distance \( d_{\text{min}} \), a bounded distance decoding algorithm can correct all received words containing \( e \) symbol errors and \( s \) symbol erasures within the constraint \( (2e + s) < d_{\text{min}} \). If the received word is within \( e \) errors and \( s \) erasures of a valid code word and \( (2e + s) < d_{\text{min}} \), then the decoder will select that code word. If the selected code word is not the code word that was transmitted, then a decoder error has occurred. If there is no code word within \( e \) errors and \( s \) erasures, where \( (2e + s) < d_{\text{min}} \), then a decoder failure is declared. If decoding is completed, the values of \( e \) and \( s \) can be obtained by comparing the received and corrected words (or, in the case of the Berlekamp-
Figure 3: MDS code decomposition for type-II HARQ protocols: code word $c_{i,3}$ is in the $(n, k, n-k+1) C_3$, code word $c_{i,1}$ is in the punctured $(\frac{n}{2}, k, \frac{n}{2} - k + 1) C_1$, and code word $c_{i,2}$ is in the punctured $(\frac{n}{2}, k, \frac{n}{2} - k + 1) C_2$.

Massey algorithm, by examining the degrees of the error and erasure locator polynomials respectively).

The bounded distance MDS type-I hybrid-ARQ protocol is defined as follows. Let $d_e$ be defined as the effective diameter of the decoding operation. The effective diameter is the maximum value of the sum $(2e + s)$ for which decoding is allowed to be completed. The effective diameter $d_e$ must thus be an integer in the range $[0, d_{\text{min}} - 1]$. Whenever $(2e + s) > d_e$, or any time a decoder failure occurs, a retransmission is requested. The effective diameter $d_e$ thus defines the balance between error correction and error detection in this type-I hybrid-ARQ system.

5.2 The Performance of the MDS Type-I Protocols Within the Framework of a Type-II Protocol

When deriving the performance of a type-II protocol, two different categories of decoding operations must be considered: those operating on newly arrived code words and those operating on code words that have caused the generation of retransmission requests. Decoding operations $D_1$ and $D_2$ fall into the former category, $D_3$ falls into the latter. The rationale for this distinction lies in the fact that the average number of errors and erasures in the code word(s) to be decoded differs between the two cases. If a code word is known to have caused the generation of a retransmission request, then the expected number of errors and erasures within the code word is higher than that for a newly received code.
word for which decoding has not yet been attempted. An effective channel model must be
developed for each of the two cases if the overall performance of the type-II protocol is to
be accurately determined.

For decoding operations $D_1$ and $D_2$, the probabilities of symbol error and erasure
are determined using information about the modulation format and the communication
channel. Figure 4 shows the channel model used in the following analysis. This model
assumes that transmitted code symbols are independent and that incorrect symbols are
equally probable. The precise values for the probabilities of symbol error $p_e$ and symbol
erasure $p_s$ are highly application dependent. For example, the case of the binary modem
used over a slowly fading code symbol interleaved channel is treated in [19]. Once $p_e$ and
$p_s$ are known, however, the following analysis can be used in most applications.

Using the values for $p_e$ and $p_s$ the probabilities of retransmission and decoder error
are determined as follows. Consider the case of an $(n, k) 2^m$-ary MDS code in a bounded
distance type-I hybrid-ARQ protocol. If only linear codes are being considered, one may
assume without loss of generality that the all-zero code word has been transmitted. Let $P_{de}^j$
be the probability that a received word is within the decoding sphere of effective diameter
d_e surrounding a code word of weight $j$. If simple error correction is to be performed

Figure 4: Channel model for the RS/HARQ system with erasure decoding
without erasure decoding, \( P_{de}^j \) takes on the value

\[
P_{de}^j = \sum_{v=0}^{\lfloor \frac{d_v}{2} \rfloor} \sum_{w=0}^{\lfloor \frac{d_w}{2} \rfloor} \sum_{x=0}^{\lfloor \frac{d_x}{2} \rfloor} \sum_{y=0}^{\lfloor \frac{d_y}{2} \rfloor} \sum_{z=0}^{\lfloor \frac{d_z}{2} \rfloor} \left( \begin{array}{c} n - j \\ v \\ w \\ x \\ y \\ z \\ \end{array} \right) \left( \begin{array}{c} j \\ j - x \\ j - y \\ (n - j - v) \left(1 - \frac{p_e}{2^{m-1}}\right) \cdot (1 - p_e)^{n-j-v} p_c^{j+v+w} \left(2^m - 1\right)^{v+w} \right) \]

This expression uses a series of counting variables to enumerate all possible received code words of length \( n \) that fall within the decoding sphere and weights them by their probability of occurrence using the channel model in Figure 4.

A similar expression can be obtained for those cases in which erasure decoding is used:

\[
P_{de}^j = \sum_{v=0}^{\lfloor \frac{d_v}{2} \rfloor} \sum_{w=0}^{\lfloor \frac{d_w}{2} \rfloor} \sum_{x=0}^{\lfloor \frac{d_x}{2} \rfloor} \sum_{y=0}^{\lfloor \frac{d_y}{2} \rfloor} \sum_{z=0}^{\lfloor \frac{d_z}{2} \rfloor} \left( \begin{array}{c} n - j \\ v \\ w \\ x \\ y \\ z \\ \end{array} \right) \left( \begin{array}{c} j \\ j - x \\ j - y \\ (n - j - v) \left(1 - \frac{p_e}{2^{m-1}}\right) \cdot (1 - p_e)^{n-j-v} p_c^{j+v+w} (1 - p_e - p_s)^{n+j-v-w} \cdot (2^m - 1)^{v+w} \right)
\]

Both Equations (6) and (7) are derived in [19].

Property 5 in Section 3 provides the weight distribution for MDS codes. If \( A_j \) is the number of code words of weight \( j \), then the probability of undetected decoder error on a single code word transmission is

\[
P_E = \sum_{j=0}^{n} A_j P_{de}^j
\]

A retransmission request will be generated whenever the received word is not within the decoding sphere surrounding the correct or any one of the incorrect code words. For the nonerasure and erasure decoding cases the following expressions result [19]:

\[
P_R = \begin{cases} 
1 - P_E - \sum_{v=0}^{\lfloor \frac{n}{2} \rfloor} \left( \begin{array}{c} n \\ v \\ \end{array} \right) p_c^v \left(1 - p_e\right)^{n-v} & \text{nonerasure decoding} \\
1 - P_E - \sum_{v=0}^{\lfloor \frac{n}{2} \rfloor} \sum_{w=0}^{\lfloor \frac{n}{2} \rfloor} \left( \begin{array}{c} n \\ v \\ w \\ \end{array} \right) \left(1 - p_e - p_s\right)^{n-v-w} p_c^v p_s^w & \text{erasure decoding}
\end{cases}
\]
The value of $P_E$ computed in Equation (8) is used for both $P_E^{(1)}$ and $P_E^{(2)}$ in Equations (3) and (4). The value of $P_R$ computed in Equation (9) is used for both $P_R^{(1)}$ and $P_R^{(2)}$.

If a code word is known to have failed in decoding attempts by itself or in combination with other code words, then the mean number of code symbol errors and erasures in the code word is higher than that indicated by the channel model in Figure 4. The increase in the channel symbol erasure and error rate must be quantified if the performance of decoding operation $D_3$ is to be accurately computed. Consider the case of an $(n, k)$ MDS code that has caused the generation of a retransmission request during a decoding attempt by a decoder with effective diameter $d_e$.

If the probability that a code symbol transmitted over a memoryless channel has been received in error is $p_e$, then the expected number of errors in a received code word $c$ of length $n$ is $np_e$ before decoding. This new $p_e$ can be written in terms of this expected value as follows

$$p_e = \frac{np_e}{n} = \frac{E\{\text{number of errors in } c\}}{\text{length of } c}$$  \hspace{1cm} (10)$$

The expected value can also be computed by weighting the number of errors $e$ by the sum of the probabilities of the error patterns of weight $e$ and then summing over all possible values of $e$.

$$E\{\text{number of errors in } c\} = \sum_{e=0}^{n} e \left[ \binom{n}{e} p_e^e (1 - p_e)^{n-e} \right]$$  \hspace{1cm} (11)$$

If it is assumed that the received code word $c$ has caused the generation of a retransmission request, then $c$ cannot have fallen within the decoding spheres of effective diameter $d_e$ surrounding the correct and incorrect code words. Let $\Omega_0$ be the summation of all terms in the above expression corresponding to error patterns that are contained within the decoding sphere surrounding the all-zero code word, i.e.,

$$\Omega_0 = \sum_{v=0}^{\frac{d_e}{2}} \sum_{w=0}^{d_e-2v} v \binom{n}{v} \binom{n-v}{w} (1 - p_e - p_s)^{n-v-w} p_e^v p_s^w.$$  \hspace{1cm} (12)$$

Let $\Omega_1$ be the summation of all terms in the above expression corresponding to error patterns that are contained within the decoding sphere surrounding nonzero code words. For code words of weight $j$ define $\Omega_1(j)$ as

14
The number of code words of weight $j$ is known (Property 5), so $\Omega_1$ can be computed as follows:

$$\Omega_1 = \sum_{j=d_{\text{min}}}^{n} A_j \Omega_1(j)$$  \hspace{1cm} (14)

By removing the terms in $\Omega_0$ and $\Omega_1$ from the right hand side of Equation (11) and dividing the result by the probability of retransmission, the probability of symbol error within $c$ given that $c$ has caused the generation of a retransmission request can be obtained as

$$P(\text{code symbol error} | \text{request}) = P'_c = \frac{1}{nP_R} (np_c - \Omega_0 - \Omega_1)$$  \hspace{1cm} (15)

The value of $P_R$ in the above expression is the probability of retransmission for the code word for its initial decoding attempt.

A similar result is obtained for the probability of symbol erasure given that a retransmission request has been generated. The above expressions are slightly modified to yield the following:

$$\Psi_0 = \sum_{v=0}^{\lfloor \frac{d_{\text{c}}}{2} \rfloor} \sum_{w=0}^{d_{\text{c}}-2v} \binom{n-v}{v} \binom{n-v}{w} (1-p_e-p_s)^{n-v-w} p_e p_s^w$$  \hspace{1cm} (16)

$$\Psi_1(j) = \sum_{v=0}^{\lfloor \frac{d_{\text{c}}}{2} \rfloor} \sum_{w=0}^{d_{\text{c}}-2v} \sum_{x=0}^{d_{\text{c}}-2v-2x} \sum_{y=0}^{d_{\text{c}}-2v-2x-y} \binom{w+y}{w} \binom{n-j}{v} \binom{n-j-v}{w} \binom{j}{x} \binom{j-x-y}{y} \binom{j-x-y}{z} (2^m-2)(2^m-1)^{y+x-j}$$
Figure 5 indicates the necessity of the preceding analysis. It is assumed that a (32, 12) Reed-Solomon code has been decomposed into a pair of (16, 12) punctured Reed-Solomon codes. The initial decoding operation ($D_1$ or $D_2$) has an effective diameter of $d_e = 4$ and the combined operation uses $d_e = 20$. The 32-ary symbols are transmitted in bit-serial form using a coherent BPSK modem over a Rayleigh fading channel with background AWGN. Erasures are generated using channel side information with an erasure threshold of $\lambda_s = 0.1$ (assuming unity energy signaling) [19], [20]. Figure 5 clearly shows that the probability of symbol error in code words that are known to have caused the generation of a retransmission request ($p'_e$) is substantially higher than that for newly arrived code words ($p_e$).

\[
p_e^{j + v - y - z} p_s^{w + y} (1 - p_e - p_s)^{n + x - j - v - w} \tag{17}
\]

\[
\Psi_1 = \sum_{j=d_{\text{min}}}^{n} A_j \Psi_1(j) \tag{18}
\]

\[
P(\text{code symbol erasure | request}) = p'_s = \frac{1}{n P_R} (n p_s - \Psi_0 - \Psi_1) \tag{19}
\]
This preceding analysis can be carried through one additional step to account for code words that have caused the generation of retransmission requests in decoding operation $D_3$ as well. The additional increase in the probability of error is small compared to the initial increase indicated by the retransmission request generated during the first decoding attempt. Therefore, the reliability and throughput calculations are tight upper and lower bounds, respectively. As will be shown in Section 6, the additional computational complexity is thus not warranted in most cases.

All of the necessary probabilities are now available for the characterization of the performance of the MDS type-II hybrid-ARQ protocol. The probabilities of symbol error and erasure from Figure 4 (first attempt to decode) and Equations (15) and (19) (second and subsequent attempts to decode) are used in Equations (8) and (9) to determine the performance of the individual type-I hybrid-ARQ protocols. These performance parameters are then used in Equations (3) and (4) to determine the overall performance of the composite type-II protocol.

6 Examples

In this section several examples of the proposed protocol are examined. Additionally, the qualitative effects of the decoding sphere sizes are considered. This section concludes with consideration of a modification of the proposed protocol that reduces the complexity of the decoder. In the following examples code symbols are transmitted in bit-serial form using a coherent BPSK modem over a code symbol interleaved Rayleigh fading channel with background AWGN. Erasures are generated using channel amplitude side information [19],[20].

In the first set of performance curves (Figures 6 and 7), a (16, 4) MDS code (code $C_3$) is decomposed into a pair of (8, 4) punctured MDS codes (codes $C_1$ and $C_2$). The original (16, 4) code and the punctured codes form a type-II HARQ protocol using the methods discussed in previous sections of this paper. Decoding operations $D_1$ and $D_2$ both have an effective diameter of $d_e = 2$, while $D_3$ has an effective diameter of $d_e = 12$. The performance of a type-I protocol ($d_e = 2$) based solely on one of the punctured (8, 4) codes has been included for reference. Figure 6 clearly shows that the type-II protocol offers substantially better throughput performance at lower signal to noise ratios. On a nonsta-
Figure 6: Throughput performance for (8,4)/(8,4) MDS type-II HARQ protocol compared to (8,4) MDS type-I HARQ protocol

Figure 7: Reliability performance for (8,4)/(8,4) MDS type-II HARQ protocol compared to (8,4) MDS type-I HARQ protocol
tionary channel, the type-II protocol thus offers more graceful throughput degradation as the channel deteriorates. Figure 7 shows an improvement in reliability performance at low signal to noise ratios. This is a direct result of the reduction in the number of transmission attempts per code word in the type-II protocol (see the denominator in Equation (4)).

![Figure 8: Throughput performance for (32,24)/(32,24) MDS type-II HARQ protocol compared to (32,24) type-I HARQ protocol](image)

In the next example a (64,24) MDS code (code C3) is decomposed into a pair of (32, 24) punctured codes (codes C1 and C2). Decoding operation D3 has an effective diameter of $d_e = 38$ while operations $D_1$ and $D_2$ have effective diameter of $d_e = 6$. The throughput data in Figure 8 indicates that the type-II protocol is still substantially better than the actual performance provided by the comparable type-I protocol. Also, as shown in Figure 9, the reliability of the type-II system is substantially better at low SNR's.

The effective diameter of the combined and single code affects both the reliability and throughput of the type-II system. Lower $d_e$ reduces the size of the decoding spheres for the punctured codes. For a type-I system this reduction increases the reliability and decreases throughput. The combined diameter, $d_{e3}$, experiences the same effect. The optimal setting for $d_e$ and $d_{e3}$ is obviously a trade-off between reliability and throughput. For maximum throughput, the decoding diameters should be as large as possible. Consider
Figure 9: Reliability performance for (32,24)/(32,24) MDS type-II HARQ protocol compared to (32,24) type-I HARQ protocol

the throughput and reliability of the (8,4) system above. The reliability and throughput at SNR of 0 dB is summarized in Table 2. The first number in each entry is the throughput, while the second is the log of the accepted word error rate. The trade-off between reliability and throughput is very clear.

Finally, consider the implications of a slight modification to the proposed protocol. After a retransmission, let the newly received code word be combined directly with the previously determined unreliable code word. This step reduces the overall processing time of the decoder. The reader is referred to Appendix C for details of the throughput and reliability calculations. For the (8,4) MDS code considered above, the direct combination approach reduces the decoder processing complexity and results in a slight throughput performance improvement. These results are shown in Figure 10. The direct combination approach is possible because of the excellent incremental redundancy available in MDS codes. The direct combination approach justifies the assumption in Section 5.2 that the modified BER for the combined decoding operation does not differ significantly from the modified BER after a single decoding operation.

20
<table>
<thead>
<tr>
<th>$d_c$</th>
<th>$d_e = 0$</th>
<th>$d_e = 2$</th>
<th>$d_e = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_{c3} = 4$</td>
<td>0.0053/-3.35</td>
<td>0.028/-1.83</td>
<td>0.126/-0.62</td>
</tr>
<tr>
<td>$d_{c3} = 5$</td>
<td>0.0057/-3.39</td>
<td>0.028/-1.83</td>
<td>0.126/-0.62</td>
</tr>
<tr>
<td>$d_{c3} = 6$</td>
<td>0.0126/-3.73</td>
<td>0.033/-1.89</td>
<td>0.127/-0.63</td>
</tr>
<tr>
<td>$d_{c3} = 7$</td>
<td>0.0142/-3.78</td>
<td>0.034/-1.91</td>
<td>0.128/-0.63</td>
</tr>
<tr>
<td>$d_{c3} = 8$</td>
<td>0.032/-4.14</td>
<td>0.047/-2.05</td>
<td>0.132/-0.65</td>
</tr>
<tr>
<td>$d_{c3} = 9$</td>
<td>0.036/-4.18</td>
<td>0.05/-2.08</td>
<td>0.134/-0.65</td>
</tr>
<tr>
<td>$d_{c3} = 10$</td>
<td>0.0697/-4.38</td>
<td>0.077/-2.27</td>
<td>0.145/-0.69</td>
</tr>
<tr>
<td>$d_{c3} = 11$</td>
<td>0.076/-4.25</td>
<td>0.082/-2.3</td>
<td>0.148/-0.69</td>
</tr>
<tr>
<td>$d_{c3} = 12$</td>
<td>0.119/-3.69</td>
<td>0.12/-2.44</td>
<td>0.168/-0.75</td>
</tr>
</tbody>
</table>

Table 2: Performance comparison of the (8,4) MDS code for various decoding sphere sizes.

![Figure 10](image)

Figure 10: Comparison of the type-II protocol and the direct combination extension.
7 Conclusion and Comments

MDS codes have been shown to exhibit a series of properties that make them well suited for use in type-II hybrid-ARQ protocols. Strong separability and strong invertibility allow for the use of a decomposition process through which an \((n, k)\) MDS code is used to create a pair of punctured \((n/2, k)\) MDS codes. The original code and the two derived codes are used individually in type-I hybrid-ARQ protocols. Together the three type-I protocols create a type-II protocol whose throughput and reliability performance is superior to that of any of the individual protocols.

The MDS decomposition process can be extended to develop more powerful code combining schemes. For example, a \((64, 4)\) MDS code can be decomposed into eight \((8, 4)\) codes to create a code combining system with eight different code rates similar to the variable-rate systems developed by Pursley and Sandberg [10]. Such a system will offer better performance than a type-II HARQ protocol in applications in which the channel varies slowly over a wide range of ambient noise levels.

A Performance Bounds on the Type-II System

Unfortunately, the complexity of Equation (7) increases with the fifth power of the effective diameter of the decoding operation. The computation of Equation (7) (as used in Equation (8)) thus begins to become a problem for the combined decoding operation \(D_3\) for code lengths of 32 or more. If sufficient computing resources are not available, the following analysis can be used to obtain bounds on the performance of the type-II system.

Consider an \((n, k)\) MDS code and a corresponding decoder with effective diameter \(d_e\). A decoding error will occur if the received word is within the decoding sphere of diameter \(d_e\) surrounding an incorrect code word. The closest such code word is Hamming distance \(d_{min}\) away (the code is assumed to be linear). There must thus be a minimum of \((d_{min} - \lfloor d_e/2 \rfloor)\) symbol errors in the received word for a decoder error to occur. If erasure decoding is available, a decoder error can occur only if, in addition to the above, the number of erasures is not greater than the effective decoding diameter \(d_e\) (otherwise a retransmission request will be generated). An upper bound is obtained by treating this pair of required events as if they were independent.
The probability of retransmission is upper bounded by the probability that \((2\epsilon + s) > d_e\).

\[
P_R \leq \begin{cases} 
1 - \sum_{v=0}^{\lfloor \frac{d_{\text{min}}}{2} \rfloor} \binom{n}{v} p_e^v (1 - p_e)^{n-v} & \text{nonerasure decoding} \\
1 - \sum_{v=0}^{\lfloor \frac{d_{\text{min}}}{2} \rfloor} \sum_{w=0}^{d_{\text{er}} - 2v} \binom{n}{v} \binom{n-v}{w} (1 - p_e - p_s)^{n-v-w} p_e^v p_s^w & \text{erasure decoding}
\end{cases}
\]  \tag{21}

The probabilities of symbol error and erasure in the above expressions \((p_e \text{ and } p_s \text{ respectively})\) are obtained from either Figure 4 (first attempt to decode) or Equations (15) and (19) (second and subsequent attempts to decode). If the code length is such that Equations (15) and (19) require unreasonable computation times, then the following approximations can be used.

\[
P(\text{code symbol error} \mid \text{request}) = p'_e \approx \frac{1}{nP_R}(np_e - \Omega_0) \tag{22}
\]

\[
P(\text{code symbol erasure} \mid \text{request}) = p'_s \approx \frac{1}{nP_R}(np_s - \Psi_0) \tag{23}
\]

where \(\Omega_0\) and \(\Psi_0\) are as in Equations (12) and (16).

**B Proof of Property 4**

Property 4 can be proved constructively as follows. Let a \(k\)-bit message \(m\) be encoded using an arbitrary generator matrix \(G\) for an \((n, k)\) MDS code \(C\). Assume (without loss of generality) that it is desired to recover \(m\) from \([c]_k\), the first \(k\) coordinates of code word \(c \in C\).
Proof

Property 3  ⇒ Any $k$ columns of $G$ are linearly independent [15]
⇒ Any $(k \times k)$ submatrix of $G$ can be reduced to an identity
  matrix through Gaussian elimination.
⇒ ∃ nonsingular $(k \times k)$ matrix $R$ such that $RG = G = [I_k|P]$
  Let $[G]_k$ be defined to be the $(k \times k)$ matrix containing
  the first $k$ columns of $G$.
⇒ $R[G]_k = [I]_k$
⇒ $[G]_k = R^{-1}$
⇒ $[c]_k = m[G]_k = mR^{-1}$
⇒ $[c]_kR = [mR^{-1}]R = m$
  QED.

C Direct combination

The direct combination extension to the proposed type-II hybrid protocol is described
by the generic graph and branch label table shown in Figure 11 and Table 3, respectively.
<table>
<thead>
<tr>
<th>Branch label</th>
<th>Throughput label</th>
<th>Reliability label</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>$T^{n_1}$</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>$p_R^{(1)} \cdot T^{n_2}$</td>
<td>$p_R^{(1)}$</td>
</tr>
<tr>
<td>c</td>
<td>$1 - p_R^{(1)}$</td>
<td>$p_E^{(1)}$</td>
</tr>
<tr>
<td>d</td>
<td>$1 - p_R^{(3)}$</td>
<td>$p_E^{(3)}$</td>
</tr>
<tr>
<td>e</td>
<td>$p_R^{(3)} \cdot T^{n_1}$</td>
<td>$p_R^{(3)}$</td>
</tr>
<tr>
<td>f</td>
<td>$p_R^{(3)} \cdot T^{n_2}$</td>
<td>$p_R^{(3)}$</td>
</tr>
<tr>
<td>g</td>
<td>$1 - p_R^{(3)}$</td>
<td>$p_E^{(3)}$</td>
</tr>
</tbody>
</table>

Table 3: Graph labels for the derivation of throughput and reliability generating functions for the direct combination extension.

The generic transfer function is as follows:

$$\text{IT} = \left\{ ac + ab \left( d + \frac{eg}{1 - ef} \right) \right\} CA. \tag{24}$$

The reliability function is easily determined as

$$P(E) = P_E^{(1)} + \frac{P_E^{(3)} P_R^{(1)}}{1 - P_R^{(3)}},$$

and the throughput is

$$\eta = \frac{k \left( 1 - P_R^{(3)2} \right)}{n_1 + n_2 P_R^{(1)} + n_1 P_R^{(1)} P_R^{(3)} - n_1 P_R^{(3)2}}.$$

The *a posteriori* or modified BER for the combined decoding operation must be averaged to account for the additional errors in the unreliable code word and the raw channel BER from the new code word.

**References**


The partial path metrics generated during Viterbi decoding provide side information that can be used in the development of a type-I hybrid-ARQ error control system. Each path through the Viterbi decoding trellis is equivalent to a path through a weighted directed graph. Error patterns in decoded information blocks thus correspond to cycles in the graph. A sliding window is used to trap these cycles. The change in partial path metric across the window determines the reliability of the maximum likelihood path. If the maximum likelihood path is deemed unreliable, a retransmission request is generated. Implementation of this error-trapping ARQ scheme requires a minor modification of the Viterbi decoder. It is shown that the reliability of the data can be greatly improved while incurring only a small reduction in the throughput. Les métriques associées aux parcours incomplets lors du décodage de Viterbi fournissent des informations complémentaires qui peuvent être utilisées pour le développement d'un système de contrôle des erreurs dans les protocoles ARQ-hybride de type I. Chaque parcours dans le treillis du décodeur de Viterbi est équivalent à un parcours pondéré dans le graphique direct. Les patrons d'erreurs dans les blocs d'information décodés correspondent aux cycles dans le graphique. Une fenêtre glissante est utilisée pour piéger ces cycles. Le changement dans la métrique du parcours incomplet dans cette fenêtre indique la fiabilité du parcours le plus vraisemblable. Si le parcours le plus vraisemblable s'avère incertain, une demande de retransmission est émise. La réalisation pratique d'un protocole ARQ à piégeage d'erreurs nécessite des modifications mineures de Viterbi. On montre que la fiabilité peut être considérablement améliorée tout en diminuant légèrement le débit des données.

Introduction

The reliability of data transmission systems can be increased through the use of error control coding. The act of coding a packet of information is simply the addition of redundant information to the packet which can be used by the receiver to detect and/or correct errors induced by the channel. Forward error correction (FEC) protocols use the redundant information exclusively for error corrections, while automatic-repeat-request (ARQ) protocols use the information solely for error detection. FEC schemes can be combined with ARQ schemes to form a protocol that provides reliability performance near that of ARQ schemes, but with throughput levels common to FEC schemes. The combination of FEC and ARQ schemes was first suggested by Wozencraft and Horstein and is known as hybrid-ARQ.

Hybrid-ARQ protocols can be grouped into two categories: type-I and type-II. In a type-I hybrid-ARQ scheme, data packets are encoded for both error detection and error correction. The receiver uses the error correction capacity to correct the most frequently occurring error patterns. The residual detection capacity is used to detect less frequently occurring error patterns. If such a pattern is detected, a retransmission request is sent back to the transmitter. Type-II hybrid-ARQ schemes differ from type-I in that the initial transmission is encoded solely for error detection. If an error is detected, the receiver requests parity bits for error correction and combines them with the original packet. If any errors are detected after error correction is attempted, then more parity bits for error correction are sent or the original packet is retransmitted. This process continues until the maximum likelihood error pattern within the received packet is correctable. At very low channel error rates the type-II schemes behave as pure ARQ schemes. They provide a higher throughput than type-I schemes because the initial type-II transmission has fewer parity bits. At high channel error rates the type-II schemes have the capability for greater error correction than type-I schemes and hence can once again achieve a higher throughput. At moderate channel error rates, however, the type-I schemes can be optimized for throughput and reliability and can outperform the type-II schemes.

This paper presents a type-I hybrid-ARQ scheme based on the Viterbi algorithm. To facilitate the comparison of the proposed scheme with other type-I hybrid-ARQ protocols, the following section describes the basic methods for implementing type-I hybrid-ARQ protocols.

Type-I hybrid-ARQ

There are two methods for implementing type-I hybrid-ARQ protocols. The first is the encoding of data packets for error detection and error correction using two separate codes. The receiver corrects as many errors as possible using the error correction code. If any errors are detected after error correction, the receiver requests a retransmission. This method can be optimized for throughput and reliability by adjusting the amount of redundancy in the error correcting and error detecting codes. But, the implementation of this scheme requires two encoders and two decoders.

An alternative method for implementing a type-I hybrid-ARQ involves modifying an FEC decoder to provide for retransmission requests. The FEC decoder is examined to find a source of side information that can be used to estimate the reliability of the decoded data. The decoder is then modified to monitor the side information during the decoding process and to determine whether a retransmission should be requested. Block codes provide several examples of such side information: consider the order of the error locator polynomial in BCH and Reed-Solomon codes and the check bits of a majority logic decoder. Similar examples are found for convolutional codes, including the time required to decode a
The type-I hybrid-ARQ scheme proposed here involves the modification of the Viterbi decoder to monitor the growth rate of the metrics during the decoding process. Since this scheme is similar to other type-I hybrid-ARQ schemes developed from modified decoders for convolutional codes, it may be useful to examine these other schemes in more detail.

Convolutional codes and type-I hybrid-ARQ schemes

The use of convolutional codes in hybrid-ARQ schemes has been the topic of considerable research. Several methods for adapting the decoders for use in type-I hybrid-ARQ schemes have been developed. These schemes each take advantage of the side information available in the decoding process to determine the reliability of the decoded data.

Sequential decoders provide at least two sources of reliability information. The time required to decode a packet using a sequential decoder is a function of the number of errors present in the received data. As the number of errors increases, the length of time required to decode the packet increases. Therefore, a natural method for generating retransmission requests is to set a limit on the time allowed to decode the packet. This is referred to as the time-out algorithm (TOA).12,13 The metric in the decoding process, the Fano metric, is also a function of the number of errors in the received sequence. During the decoding process, the rate of change of the Fano metric can be monitored. If the rate of change of the metric exceeds a predefined threshold, then the packet is declared unreliable and a retransmission is requested. This scheme is known as the slope control algorithm (SCA).12,13

Majority-logic decoding is an algebraic technique used in the decoding of both block and convolutional codes.22 The decoder generates J orthogonal checksums for each data bit. The checksums “vote” to determine whether each data bit is in error. For example, let \( J \) be the number of checksums equal to 1. A measure of the reliability of the decoding process is the extent of the majority among the checksums. A threshold \( r \) on the majority logic circuit is used to check the reliability of each bit error decision.18

If \( \eta \leq \left\lfloor \frac{J}{2} \right\rfloor - \left\lfloor \frac{\tau}{2} \right\rfloor \), then no data bit error is assumed.

If \( \eta \geq \left\lfloor \frac{J}{2} \right\rfloor + \left\lfloor \frac{\tau}{2} \right\rfloor + 1 \), then the data bit is declared in error.

If \( \left\lfloor \frac{J}{2} \right\rfloor - \left\lfloor \frac{\tau}{2} \right\rfloor < \eta < \left\lfloor \frac{J}{2} \right\rfloor + \left\lfloor \frac{\tau}{2} \right\rfloor + 1 \), then the majority logic decision is declared unreliable and a retransmission is requested.

The Viterbi decoder has also been modified for use in type-I protocols. In the Viterbi algorithm, each codeword can be thought of as a path through a trellis diagram. The path which corresponds to the maximum likelihood transmitted codeword is identified through the calculation and comparison of path metrics. Yamamoto and Itoh proposed the first modification of the Viterbi decoder for use in a type-I hybrid-ARQ protocol.17 If the difference between the path metric of the surviving path at each node and that of the closest non-survivor do not exceed a given threshold, then the surviving path is declared to be unreliable. If all paths are declared to be unreliable, then a repeat request is generated. Wicker also proposed a repeat request algorithm based on the Viterbi decoder.

This paper presents a third modification of the Viterbi algorithm for use in type-I hybrid-ARQ systems. The path metrics calculated in the decoder are used to “trap” some of the less frequently occurring error patterns using a sliding window. By monitoring the rate of increase of the partial path metric across this window, errors are detected. If the rate of increase exceeds a given threshold, then it is assumed that too many errors have occurred to reliably decode the sequence. This method offers a simple yet powerful alternative to those proposed by Yamamoto and Itoh and Wicker.

The Viterbi decoder and the trap algorithm

A convolutional encoder can be described using a state diagram. The state diagram is extended in time to form a trellis diagram. For each distinct input sequence \( \bar{d} \), there is a unique path through the trellis diagram and a unique output codeword \( \bar{x} \). When a sequence \( \bar{r} \) is received, the Viterbi decoder determines which path through the trellis diagram is the one most likely to correspond to the transmitted codeword. The Viterbi algorithm accomplishes this by translating the decoding problem into that of finding the maximum likelihood path through a weighted directed graph.24

For each path through the trellis diagram a path metric \( M(\bar{r} | \bar{x}) \) is calculated. \( M(\bar{r} | \bar{x}) \) is a function of the log of the probability of \( \bar{r} \) being received given that \( \bar{x} \) was transmitted (i.e. \( \log P(\bar{r} | \bar{x}) \)). For a memory-less channel, the path metric is simply the sum of the individual bit metrics \( M(r_i | x_i) \). If the channel is also binary then

\[
M(r_i | x_i) = \begin{cases} 
0 & \text{if } r_i = x_i \\
1 & \text{if } r_i \neq x_i 
\end{cases}
\]

The Viterbi decoder chooses the path and the corresponding \( \bar{x} \) which has the lowest path metric. This corresponds to the maximum likelihood transmitted codeword.

To simplify the analysis, consider the case of codeword transmission over the binary symmetric channel. The path metrics for this channel correspond to the Hamming distance between the received sequence and the codewords. The minimum Hamming distance between codewords is called the minimum free distance \( d_{\text{free}} \). A decoding error can thus occur only if there are \( \lfloor d_{\text{free}}/2 \rfloor \) or more errors in the received sequence. Since convolutional codes are linear, it can be further assumed without loss of generality that the all zeroes codeword was transmitted. An error event is thus equivalent to a path in the trellis diagram which departs from the zero state and returns to the zero state at a future time. The received sequence must have a weight of \( \lfloor d_{\text{free}}/2 \rfloor \) or more in order for the decoder to incorrectly decode the sequence.

The error-trapping ARQ scheme presented here monitors the growth in the path metric for each codeword over a specific interval (window). The number of branches in the window is called the “transition trap length” \( \Gamma \). Consider all paths which merge with the all zero path at a particular node in the trellis. \( \Gamma \) is selected to equal the minimum number of branches over which all such paths will have a Hamming weight greater than or equal to \( d_{\text{free}} \). By linearity, any path which merges with any codeword will thus be at least a Hamming distance \( d_{\text{free}} \) from the codeword over the last \( \Gamma \) branches. Let \( M_{\text{free}} \) be the path metric of the surviving path at node \( k \) after the \( k \)th transition of the trellis diagram. The rate of increase of the path metric over the previous \( \Gamma \) branches is defined as \( M_{\text{free}}(k) = M_{\text{free}}(r) - r \), where \( M_{\text{free}}(r) \) is the partial path metric of the surviving path traced \( \Gamma \) branches back. If the rate of increase of the path metric exceeds a fixed retransmission threshold \( r \), then the path is considered
The error-trapping ARQ algorithm presented is similar to the ARQ modification of the Viterbi algorithm proposed by Wicker. Wicker's algorithm used a threshold on the partial path metric up to the current time. A sliding window was not used, but rather a limit was placed on the total number of bit errors allowed in the entire packet. The error-trapping ARQ algorithm will result in a lower retransmission probability than the Wicker modification.

Yamamoto and Itoh's ARQ algorithm differs from the error-trapping algorithm in that the Yamamoto and Itoh algorithm is an error event algorithm and the error-trapping algorithm is a bounded distance decoder. Of the two, the error-trapping algorithm is a natural choice when considering the modification of a Viterbi decoder that has been designed to use a traceback algorithm. The modification of the traceback algorithm for use in monitoring the growth rate of the partial path metrics is straightforward. The performance of the two hybrid-ARQ algorithms is comparable.

To illustrate the error-trapping ARQ algorithm, consider the following example. Figure 1 contains the state diagram and corresponding trellis diagram for a (2, 1, 2) convolutional code with \( d_{	ext{free}} = 5 \). A computer search indicates that the transition trap length \( \Gamma \) for this code is 6. Let \( \tau = 2 \) and assume the received sequence is \( r = (11, 10, 11, 00, 10, 01, 00, \ldots) \). The decoding proceeds as shown in Figure 2. After the first transition (\( i = 1 \)), the partial path metric through node 00 is 2 and that through node 10 is 0. Since the partial path metric through node 00 at \( i = 1 \) equals \( \tau \), all paths that pass through node 00 at \( i = 1 \) are eliminated. The process continues with all paths whose metrics have increased by \( \tau = 2 \) or more in \( \Gamma = 6 \) or less transitions being eliminated. At \( i = 7 \), the remaining two paths have a partial path metric of 2. The partial path metrics at \( i = 7 - 6 = 1 \) for these two paths are equal to 0. Since \( 2 - 0 \geq \tau \), these two paths are eliminated and a retransmission is requested. If the seventh received pair had been 01 instead of 00, then there would have remained a path through node 11 at \( i = 7 \) and the decoding would have continued.

**Performance**

The performance of the Viterbi algorithm with the error-trapping ARQ is best measured by the probability of decoder error and the expected throughput. These functions can be calculated as a function of the probability of the channel bit error rate and the retransmission threshold \( \tau \).

**Probability of error**

The Viterbi decoder with error-trapping ARQ can only make a decoding error if the maximum likelihood path is not equal to the transmitted sequence and the rate of growth of the path metric for the maximum likelihood path does not equal or exceed \( \tau \) over any \( \Gamma \) successive branches. Two different bounds on the probability of error are derived in the following sections. The first is a union bound approach which uses the generating function of the convolutional code to identify all possible error events and bounds the probability of each event. The second bound is a geometric bound which closely follows the derivation of probability of error for FEC convolutional codes by Lin and Costello. The analyses which follow assume a binary symmetric channel (BSC) with bit error probability \( p \).

**Union bound**

Define an error event after the \( l \)th transition of the Viterbi decoder as the elimination of the all zero path at the \( l \)th transition in favor of a non-zero path which has not been terminated by the trap algorithm. The length \( k \) of the error event is defined as the length of the non-zero path from the point where it diverged from the all zero path until it remerged at the \( l \)th transition of the packet. By definition, any non-zero path must have Hamming weight greater than or equal to \( d_{	ext{free}} \) and must be one of the paths enumerated by the code generation function. Finally, the length of the error event must be less than or equal to \( l \).

An error event can only occur if the received vector meets certain conditions over the length of the error event. If the length of the error event is less than or equal to \( \Gamma \) transitions, then the received vector must be within \( \tau \) bits of the error path. If the length of the
Error events of length up to $\Gamma$

Consider an error event $Y$ corresponding to a non-zero path of weight $i$ and length $k \leq \Gamma$ transitions. For a rate $\frac{m}{n}$ code over $n$ the BSC, the probability of the error event is the probability that a received vector $R$ of length $n\Gamma$ bits is less than $\tau$ bits distance from a codeword of weight $i$. Using a simple counting approach, the probability of decoder error is shown to be

$$P^\Gamma_{nj} = \sum_{i=0}^{n} \sum_{r=0}^{i} \left( \begin{array}{c} i \\ r \end{array} \right) p^{i-r} (1-p)^{n \Gamma + i - r}$$  \hspace{1cm} (2)

where $p$ = probability of channel bit error.

Error events of length greater than $\Gamma$

An error event $Y$ of length greater than $\Gamma$ can only occur if the received vector $R$ is within $\tau$ bits of the non-zero path over every $n\Gamma$ consecutive bits of the path. To derive the exact expression for the probability of this occurrence we would require knowledge of the location of each non-zero bit in the error event. Since this knowledge is not easily acquired, a bound on the probability of this error event is derived.

Let $d^{min}_\Gamma$ be the minimum weight among the non-zero paths of length $\Gamma$ transitions converging with the all zero path after the $i$th transition (by definition equal to $d_{ij}^r$). Let $d^{max}_\Gamma$ be the maximum weight among the non-zero paths of length $\Gamma$ transitions converging with the all zero path after the $i$th transition. These values can be found by a computer search. Using $d^{min}_\Gamma$ and $d^{max}_\Gamma$, a bound on the probability of any error event longer than $\Gamma$ transitions can be derived.

Consider a non-zero path of weight $i$ and length $k \geq \Gamma$ transitions. The weight of the non-zero path over the last $\Gamma$ transitions must be greater than or equal to $d^{min}_\Gamma$ bits. The weight of the non-zero path over the first $k - \Gamma$ transitions must be greater than or equal to $i - d^{max}_\Gamma$ bits. An error event can thus only occur if $R$ is within $\tau$ bits of the non-zero path over the last $\Gamma$ transitions and within $\frac{i}{2}$ bits of the non-zero path over its entire length. The probability of $Y$ occurring is thus bounded by

$$\Pr \{ Y \} \leq p^\Gamma_{nj} \cdot P^{(k-\Gamma)}_{n/2 \cdot j} \cdot d^{max}_\Gamma$$  \hspace{1cm} (3)

Probability of bit error

The non-zero paths for a convolutional code can be enumerated using a generating function. The generating function is of the form

$$T(X, Y, Z) = \sum_{i=d^{min}_\Gamma}^{\infty} \sum_{j=1}^{n} \sum_{k=1}^{\infty} A_{i,j,k} X^i Y^j Z^k$$  \hspace{1cm} (4)

where $A_{i,j,k}$ is the number of paths of weight $i$ bits, length $k$ transitions, and $j$ non-zero information bits. Using the generating function and the bounds on probability of error events derived in the previous section, bounds on the probability of bit error for the error-trapping decoder can be derived.

Consider the decision process after the $i$th transition of a packet. The error events which are possible must all be less than or equal to $\tau$ in length. Therefore, the probability of an error occurring after the $i$th is the sum of the probabilities of all possible error events less than $\tau$ bits distance of the non-zero path over every $\Gamma$ consecutive transitions of the non-zero path.

The probability of decoder error is bounded by

$$P_E(l) = \sum_{k=1}^{\Gamma} \sum_{i=1}^{nk} B_{i,k} \cdot \left\{ \begin{array}{ll} p^\Gamma_{nj} & , k \leq \Gamma \\
 p^\Gamma_{nj} \cdot P^{(k-\Gamma)}_{n/2 \cdot j} \cdot d^{max}_\Gamma & , k > \Gamma \end{array} \right.$$  \hspace{1cm} (5)

where

$$B_{i,k} = \sum_{j=1}^{mk} A_{i,j,k}$$  \hspace{1cm} (6)

A union bound can be used to get an upper bound on the probability of decoder error. For a packet of length $N$ bits, the probability of decoder error is bounded by

$$P_E \leq \sum_{l=1}^{N/n} P_E(l)$$  \hspace{1cm} (7)

$$\leq \sum_{l=1}^{N/n} \sum_{k=1}^{\Gamma} \sum_{i=1}^{nk} B_{i,k} \cdot \left\{ \begin{array}{ll} p^\Gamma_{nj} & , k \leq \Gamma \\
 p^\Gamma_{nj} \cdot P^{(k-\Gamma)}_{n/2 \cdot j} \cdot d^{max}_\Gamma & , k > \Gamma \end{array} \right.$$  \hspace{1cm} (8)

The number of bit errors in a single error event is equal to the number of non-zero information bits corresponding to the event. By weighting each error event by the appropriate number of non-zero information bits, an upper bound on the expected number of bit errors is computed. The probability of bit error is then upper bounded by dividing by the number of information bits in each packet:

$$R_b \leq \frac{n}{mN} \sum_{l=1}^{N/n} \sum_{k=1}^{\Gamma} \sum_{i=1}^{nk} \sum_{j=1}^{mk} j A_{i,j,k} \cdot \left\{ \begin{array}{ll} p^\Gamma_{nj} & , k \leq \Gamma \\
 p^\Gamma_{nj} \cdot P^{(k-\Gamma)}_{n/2 \cdot j} \cdot d^{max}_\Gamma & , k > \Gamma \end{array} \right.$$  \hspace{1cm} (9)

The computation of this bound is simplified by rearranging the summations. Since the probability of a particular error event is independent of its location, the sum of the probability of its occurrence in each $P_E(l)$ is equal to the number of locations in the packet that this error event can occur weighted by its probability of occurrence. Since an error event of length $k$ can occur in $\frac{N}{n} - k$ locations within the packet, the probability of bit error bound can be rewritten as

$$R_b \leq \frac{n}{mN} \sum_{k=1}^{\Gamma} \left( \frac{N}{n} - k + 1 \right) \sum_{i=1}^{nk} \sum_{j=1}^{mk} j A_{i,j,k} \cdot \left\{ \begin{array}{ll} p^\Gamma_{nj} & , k \leq \Gamma \\
 p^\Gamma_{nj} \cdot P^{(k-\Gamma)}_{n/2 \cdot j} \cdot d^{max}_\Gamma & , k > \Gamma \end{array} \right.$$  \hspace{1cm} (10)

**Geometric bound**

When the generating function for the convolutional encoder is not known, the probability of error can still be bounded using the minimum free distance of the code. The following analysis employs a geometric technique for deriving an upper bound on the probability of error. This bound is not as tight as the union bound derived above, but may be useful for very complex codes for which the generating function cannot be easily derived.

An error event is defined as the occurrence of a received sequence which is less than $\tau$ bits distance from an incorrect codeword over any $\Gamma$ consecutive branches. Let $n$ be the number of bits at the
The probability of bit error can be derived using an analysis similar to that done by Lin and Costello. Assume without loss of generality that the all zero codeword was used to upper bound the probability of packet error by a factor of $2d-1$. If $d_{free}$ is the number of non-zero information bits among all codewords of weight $d_{free}$, and $k$ is the number information bits per branch, then

The probability of bit error $P_e$ is bounded as follows:

$$P_b \leq \frac{B d_{free} 2 \Delta_m p^{d_{free}}}{k}. \quad (23)$$

**Throughput**

The improvement in reliability performance offered by the proposed scheme does not come without a price. The cost of the increased reliability is a decrease in throughput caused by the retransmissions. Throughput $T$ is defined as the rate at which data is successfully transmitted across the channel. The throughput for selective-repeat protocols with infinite buffer length can be expressed as

$$T = \frac{m}{n} P_s, \quad (24)$$

where $m$ is the number of inputs to the convolutional encoder, $n$ is the number of outputs from the convolutional encoder $(c = \frac{m}{n})$, and $P_s$ is the probability of successful transmission of each packet.

An exact expression for the probability of successful transmission can be derived using combinatorial methods. The first retransmission request can occur after $\Gamma$ transitions. The probability of successfully receiving the packet up to this point is the probability that there are less than $\tau$ errors in the first $\Gamma$ bits. This probability can be written as follows:

$$P_{s0} = \sum_{j=0}^{\tau-1} \left( \frac{n!}{j!} \right) p^j (1 - p)^{n-j}. \quad (25)$$

At the $(\Gamma + k)$th transition $(k > 0)$, the probability of successful reception becomes a conditional probability. Assume that the packet was successfully received up to the current transition. It is thus assumed that there are less than $\tau$ errors in the $\Gamma$ bits prior to the $(\Gamma + k)$th transition. The probability of successful reception $P_{sk}$ thus becomes

$$P_{sk} = \frac{\min(n, \tau - 1)}{P_{s0}} \left[ \text{Prob.} \{i \text{ errors in transition} \} \cdot P_{sk} \right]. \quad (26)$$

where

$$\text{Prob.} \{i \text{ errors in transition} \} = \left( \frac{n!}{i!} \right) p^i (1 - p)^{n-i}. \quad (27)$$

and

$$P'_{sk} = \text{Prob.} \{< \tau - i \text{ errors in prior } n \Gamma - n \text{ bits} \}$$

$$= \frac{1}{\text{Prob.} \{< \tau \text{ errors in } n \Gamma \text{ bits} \}} \left[ \sum_{j=0}^{\tau-1} \left( \frac{n!}{n-j!} \right) \sum_{m=j}^{\min} \left( \frac{m!}{m-j!} \right) p^m (1 - p)^{n-m-j} \right]. \quad (28)$$

$$= \frac{1}{P_{s0}} \sum_{j=0}^{\tau-1} \left( \frac{n!}{n-j!} \right) \sum_{m=j}^{\min} \left( \frac{m!}{m-j!} \right) p^m (1 - p)^{n-m-j}. \quad (29)$$
The value of $P_{sk}$ is constant for $k > 0$. The total probability of successful packet transmission is thus

$$P_s = P_{sk}^n (n/n)^{\gamma-n}.$$

(32)

Performance example

A simulation of a Viterbi decoder with error-trapping ARQ was developed to verify the analysis of the algorithm. The simulation sent all zero packets with a length of 1000 bits over a binary symmetric channel. A total of 1000 packets were sent over the channel for each channel bit error rate simulated. The simulation throughput was calculated for comparison with the derived throughput.

Consider a convolutional code with the following parameters:

- Rate = $\frac{1}{2}$
- $d_{free} = 6$
- Memory length = 3.

The generating function for the selected code is

$$T(X, Y, Z) = \frac{X^6Y^2Z^5 + X^7Y^4Z^4 - X^8Y^2Z^5}{\Delta},$$

(33)

where

$$\Delta = 1 = XY(Z + Z^2) - X^2Y^2(Z^4 - Z^3) - X^4Y^2(Z^3 - Z^4) - X^6Y^2Z^5 - X^6Y^2Z^5.$$ 

(34)

A computer search revealed the following error-trapping ARQ parameters:

- $\gamma = 9$
- $d_{min} = d_{free} = 6$
- $d_{max} = 13$.

This code is transmitted over a binary symmetric channel. The threshold $\tau$ is allowed to vary between 1 and $\frac{d_{free}}{2} = 3$, and the packet length is 1000 bits. Note that a threshold of $\tau = 1$ is equivalent to a pure ARQ system since the detection of a bit error causes the path metric to increase by one and the maximum likelihood path is declared unreliable. Since the generating function for this code is available, the union bound on the probability of error can be calculated. Figure 3 shows the union and geometric bound of the probability of decoder error for the error-trapping ARQ system as a function of the channel bit error rate. Note that the error-trapping ARQ modification can greatly improve the reliability of the transmitted data for small values of $\tau$. Figure 4 shows the expected throughput of the error-trapping ARQ system as a function of channel bit error rate. Also included in Figure 4 is the simulated throughput using the same code. Note that for the case of $\tau = 2$, the increased reliability comes at a cost of only a modest decrease in throughput for channel bit error rates less than 0.005. Note also that the simulated and calculated throughput for this code are almost exactly the same, as expected.

Simulation results have shown that the error-trapping system's performance is comparable to the performance of the Yamamoto and Itoh algorithm. Both algorithms were simulated using the (2, 1, 3) convolutional code over a channel with a bit error rate of 0.01 and a packet length of 1000 bits. A sufficient number of packets were sent over the channel to insure that at least 15 bit errors were generated by the decoder. The error-trapping system with threshold $\tau = 3$ had a throughput of 0.451 and a decoded bit error rate of 3.55e-7. The Yamamoto and Itoh Algorithm with threshold $\lambda = 2$ had a throughput of 0.480 and a decoded bit error rate of 6.25e-7. The throughput of the Yamamoto and Itoh algorithm was slightly higher than that of the error-trapping system, but also had a slightly higher decoded bit error rate. Based on this and other data points,
Consider another convolutional code with the following parameters:

- Rate = \( \frac{1}{2} \)
- \( d_{\text{free}} = 10 \)
- Memory length = 6.

This code is far more complex than that in the first example. Calculating the generating function for this code requires finding the transfer function of a state diagram with 64 states. Therefore, the geometric bound for the probability of error is used for this code. A computer search reveals that the transition trap length \( r \) for this code is 26. The threshold \( r \) is allowed to vary between 1 and \( \frac{d_{\text{free}}}{2} \), and the packet length is 1000 bits. Figure 5 shows the geometric bound of the probability of decoder error for the error-trapping ARQ system as a function of the channel bit error rate. The upper bound of the FEC error probability is determined by letting \( r = \left\lfloor \frac{d_{\text{free}} + 1}{2} \right\rfloor \). Figure 6 shows the expected throughput of the error-trapping ARQ system as a function of channel bit error rate. Also included in Figure 6 is the simulated throughput using the same code. Again, the simulated and calculated throughput for this code are almost exactly the same.

As an example of the capability of the error-trapping ARQ scheme, consider the above code with a threshold \( r = 3 \) over a channel with a bit error rate of \( 10^{-3} \). From Figure 5, the probability of decoder error using the error-trapping ARQ scheme is less than \( 10^{-12} \). Figure 6 shows that the throughput using the error-trapping scheme is still greater than 0.45 (as compared to 0.5 for the FEC case). The unmodified Viterbi decoder over the same channel has a probability of decoder error less than \( 10^{-6} \). The error-trapping ARQ modification produced a significant increase in the reliability of the transmitted data with only a modest reduction in the throughput.

**Conclusion and applications**

The Viterbi decoder can be modified for use in type-I hybrid-ARQ systems by simply storing the partial path metrics and implementing a traceback algorithm. The traceback algorithm recovers the partial path metric \( r \) branches back for each active path. This is compared with the current partial path metric to determine if the rate of increase of the path metric exceeds a given threshold \( r \). Any path whose rate exceeds \( r \) is terminated. If all paths are terminated, then a retransmission of the packet is requested. This simple modification greatly increases the data reliability at the expense of only a slight decrease in throughput over a wide range of channel bit error rates. The modified decoder is well suited for applications where a high level of data reliability is needed.

Type-I hybrid-ARQ schemes such as the error-trapping ARQ algorithm have a number of possible applications. For example, the error-trapping ARQ algorithm can be used over the non-stationary channels seen in many applications. The threshold \( r \) is varied to compensate for changes in channel bit error rates. In this manner, the reliability of the channel can be held constant while maintaining the highest allowable throughput.

The error-trapping ARQ algorithm can also be used to develop adaptive error control systems based on diversity and code combining schemes. In conventional ARQ or hybrid-ARQ systems, the receiver discards any packet for which a retransmission is requested. Only the most recently received packet is decoded and all information from previously sent packets is disregarded. Diversity and code combining schemes take advantage of all the information available by using all received copies to estimate the transmitted data bits. Several schemes for doing this have been proposed in recent publications. Chase proposes using a Viterbi decoder in a code combining scheme which combines an arbitrary number of packets. Chase's scheme is a type-I hybrid-ARQ scheme using a separate error detection code and convolutional error correction code. The error-trapping ARQ algorithm proposed in this paper could be used in place of the dual-code type-I hybrid-ARQ scheme proposed by Chase. This eliminates the need for two encoders and decoders and increases the throughput by eliminating the need for extra parity bits for error detection.

The error-trapping ARQ algorithm can also be used in a Viterbi code combining scheme in which copies of the original received packet are interleaved with the original copy to form an increasingly lower rate code. Suppose that a rate 1/2 convolutional code with minimum distance \( d_{\text{free}} \) is transmitted, and that the receiver uses the...
...to copy a packet, then a second copy of the packet is sent and is interleaved with the received copy of the first packet at the receiver. The resulting code is a rate 1/4 code with a minimum distance of 2d_{min}. The threshold \( r \) can now be adjusted in such a way as to maintain the same probability of decoded bit error. The decoder is still a maximum-likelihood decoder, and because of the reduced rate, the combined codeword would have a greater probability of being decoded successfully. If additional retransmissions are requested for this packet, the method can be extended to combine packets to form codewords of rate 1/6, 1/8, or less. In this manner, the rate of the code adjusts to changes in the channel conditions and greater throughput is achieved over channels with variable error rates.

References

The design and implementation of type-I and type-II hybrid-ARQ protocols based on first-order Reed-Muller codes

Stephen B. Wicker and Michael Bartz
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Abstract

It is shown that the Green machine decoder can be modified to allow for the use of first-order Reed-Muller codes in both type-I and type-II hybrid-ARQ protocols. The type-I protocols offer substantially better reliability performance than their FEC counterparts, though at the expense of a reduction in throughput at low SNR’s. The type-II protocols offer similar reliability performance to that provided by the the type-I protocols, but offer much better throughput performance at low SNR’s. Both the type-I and type-II protocols are obtained through minor modifications of the Green machine that do not appreciably increase the complexity or decrease the speed of the decoder.

1 Introduction

Reed-Muller (RM) codes comprise one of the oldest and best understood families of error correcting codes [1-7]. Though they can best be characterized as “weak codes that are easy to decode” [2], RM codes have seen wide application and can certainly claim to be well traveled. For example, the first-order RM code of length 32 was used to control telemetry link errors on all of the Mariner deep-space probes flown between 1969 and 1977 [3]. The use of the first-order code was facilitated by the development of a highly efficient maximum likelihood decoding algorithm by R. R. Green [3]. This “Green machine” uses a particularly fast implementation of the Hadamard transform, a close relation to the Discrete Fourier Transform. In

\[\text{This work was supported by the National Science Foundation under Grant Number NCR-9009877.}\]
This paper methods are demonstrated for modifying the Green machine for use in type-I and type-II hybrid-ARQ protocols. The resulting systems provide improved throughput and reliability performance at extremely low signal-to-noise ratios ($\leq 0$ dB). The only requirement placed on the transmitters in the proposed systems is that they be able to retransmit selected data packets on command. The techniques discussed in the following pages are thus readily applied to many existing systems in which the transmitter is no longer available for modification.

In Section 2 a brief review of first order Reed-Muller codes and the Green machine provides a basis for hybrid-ARQ protocols discussed later. In Section 3 a Reed-Muller type-I hybrid-ARQ system is developed that is based upon reliability information available from the Green machine. A type-II protocol system based upon coding combining is discussed in Section 4 and the simple design modifications to a Green machine decoder are presented.

## 2 First-order Reed-Muller codes

The first order RM code $R(1, m)$ of length $2^m$ is defined through its association with the set of all Boolean functions in $m$ variables of degree zero and one. The code words of $R(1, m)$ are the truth tables formed by the functions $f(v_1, v_2, \cdots, v_m)$ as they are evaluated over all binary $m$-tuples in their natural ascending order [1].

For the case of the first-order codes of length $2^m$, the minimum distance is $d_{\text{min}} = 2^{m-1}$ and the dimension $k = (m + 1)$. The weight distribution of the first-order RM codes is known. Let $A_j$ be the number of code words of weight $j$ in $R(1, m)$. It can be shown [1] that the non-zero weight enumerators are

$$A_0 = A_2^m = 1, A_{2^m-1} = 2^{m+1} - 2. \quad (1)$$

Maximum likelihood decoding of first-order RM codes is obtained through the use of the Hadamard Transform [1],[3]. Let $r$ be the binary vector at the output of the receiver bit decision circuitry. The received word $r$ is first converted into a vector containing +1's and -1's.

$$r = (r_1, r_2, \cdots, r_{2^m}) \rightarrow F = ((-1)^{r_1}, (-1)^{r_2}, \cdots, (-1)^{r_{2^m}}) \quad (2)$$

The Hadamard transform of $F$ is then computed through the matrix multiplication

$$\hat{F} = (\hat{F}_1, \hat{F}_2, \cdots, \hat{F}_{2^m}) = FH_{2^m}, \quad (3)$$

2
where $H_n$ is a specific Hadamard matrix of order $n$. (The Green machine provides a very fast and elegant means of accomplishing the computation in Equation (3). The technique used is discussed in Section 4.)

The values assumed by each of the coordinates in the Hadamard transform of $F$ can be interpreted geometrically. Consider the subset of Boolean functions that are associated with $R(1, m)$ and also have the form $f_a(v_1, \cdots, v_m) = 0 \cdot 1 + a_1v_1 + \cdots + a_mv_m$, where $a$ is the decimal equivalent of the binary number $a_1a_2\cdots a_m$. The natural ordering of the binary $m$-tuples $a_1a_2\cdots a_m$ imparts an ordering on this subset of Boolean functions $\{f_a\}$. If the received word $r$ is associated with the Boolean function $f$, then it can be shown that the $a^{th}$ coordinate of the Hadamard transform of $F$ is equal to the number of $0$'s minus the number of $1$'s in the truth table of the Boolean function $(f + f_a)$. It follows that

$$\hat{F}_a = 2^m - 2d(f, f_a)$$

where $d(f_i, f_j)$ is the Hamming distance between the truth tables associated with the functions $f_i$ and $f_j$. The decoder thus performs its function by finding the coordinate of $\hat{F}$ with the greatest magnitude. If the selected coordinate $\hat{F}_j$ is positive, then the code word associated with the function $f_j$ is selected as the transmitted code word. If $\hat{F}_j$ is negative, then the code word associated with the function $(1 + f_j)$ is selected.

3 The Reed-Muller type-I hybrid-ARQ protocol

In a type-I hybrid-ARQ protocol, packets are encoded for both error correction and error detection. The error correction capacity is designed to correct the error patterns most frequently appearing on the forward communication channel. The detection capacity is used to detect the less frequently occurring error patterns. When such patterns are detected, a retransmission request is sent to the transmitter by way of a feedback channel.

Both the error correction and the error detection functions in a type-I protocol can be implemented with the same encoder/decoder pair through the identification of reliability information generated by the decoder [8]. The reliability information is used to estimate the quality of decoded data packets and to request the retransmission of those that are deemed unreliable. One approach is to use the number of errors corrected by the decoder in each packet as a rough measure of reliability [8],
[9]. By limiting the allowed number of errors corrected to some value less than the total error correction capability of the code, reliability is increased at the expense of a reduction in throughput caused by retransmissions.

The Green machine provides a ready source of reliability information in the coordinates of the Hadamard transform of the received packets. Recall that the decoder selects the code word or the one's complement of the code word associated with the transform coordinate with the greatest magnitude. Let \( r \) be the transmitted packet and \( c_j \) the selected code word. The magnitude of the associated transform coordinate is equal to the number of 0's minus the number of 1's in the vector \( (r + c_j) \). Because the number of 1's in this vector is equal to the number of errors corrected by the decoder, the desired reliability metric is obtained by solving for \( d(f, f_a) \) in Equation (4). The number of errors corrected during the decoding of \( r \) onto \( c_j \) is thus

\[
t_r = \frac{1}{2}(2^m - |\hat{F}_j|).
\]  

(5)

A Reed-Muller type-I hybrid-ARQ protocol can now be defined as follows.

1. The receiver computes the Hadamard transform of the received packet \( r \).

2. The transform coordinate with the greatest magnitude is identified. The number of errors \( t_r \) to be corrected is then computed using Equation (5). If \( t_r > t_{\text{max}} \), the maximum allowable number of errors to be corrected, then a retransmission of the packet is requested.

3. If \( t_r \leq t_{\text{max}} \), then the code word (or its one's complement) associated with the transform coordinate with maximum magnitude is selected as the maximum likelihood transmitted code word and the corresponding data is sent to the data sink.

This protocol can be implemented by adding a simple threshold device at the end of the final Hadamard transform computation stage in the Green machine architecture. The resulting increase in complexity caused by this retransmission request generator is negligible. Additional buffering may be necessary in the transmitter and receiver, however, depending upon the retransmission protocol selected

---

1 In the two hybrid-ARQ protocols discussed in this paper it is assumed that a packet consists of a single RM code word. The analysis is modified slightly for those cases in which packets contain several code words (see [8]).
(e.g. selective-repeat (additional transmitter and receiver buffering), go-back-N (additional transmitter buffering), stop-and-wait (no additional buffering)).

3.1 Reliability and throughput performance

In this section the reliability and throughput performance of the RM type-I hybrid-ARQ protocol is derived for the code $R(1, m)$. It is assumed that $t_{\text{max}}$, the maximum number of errors allowed to be corrected, is less than or equal to $\lceil (d_{\text{min}} - 1)/2 \rceil = 2^{(m-2)} - 1$. The error correcting regions in the space of all possible received vectors are thus $2^m$-dimensional Hamming spheres of radius $t_{\text{max}}$. A decoder error occurs whenever a received packet falls within a decoding sphere surrounding a code word other than that which was transmitted. A retransmission request is generated whenever the received packet falls into the space between the decoding spheres.

The forward communication channel is assumed to be binary symmetric with crossover probability $p$. The feedback channel is assumed to be noiseless. Because the RM codes discussed here are linear, it may be assumed without loss of generality that the all-zero code word has been transmitted. The probability that the received word is exactly Hamming distance $k$ from a code word of weight $j$ is [6, p. 22]

$$P_k^j = \sum_{r=0}^{k} \binom{j}{k-r} \binom{2^m - j}{r} p^{j-k+2r}(1-p)^{2^m-j+k-2r}, \quad (6)$$

where the relationship between $j$ and $k$ must satisfy the following conditions:

$$j - k + 2r \geq 0 \quad \text{and} \quad 2^m - j + k - 2r \geq 0.$$ 

The probability of decoder error on a given transmission is then

$$P_e = \sum_{j=1}^{2^m} A_j \sum_{k=0}^{t_{\text{max}}} P_k^j$$

$$= (2^{m+1} - 2) \sum_{k=0}^{t_{\text{max}}} P_k^{2^m-1} + \sum_{k=0}^{t_{\text{max}}} P_k^{2^m}, \quad (7)$$

The word error rate among the accepted packets is

$$P(E) = \frac{P_e}{1 - P_r}, \quad (8)$$
where $P_r$ is the probability that a given transmission will cause the generation of a retransmission request [10]. The probability of bit error is determined by counting the total number of 1's in the radix-2 representation of all the information words. It is easily shown that the number of 1's in the radix-2 representation of all length $m$ binary words is $m2^{m-1}$. If the RM code is non-systematic [1, p. 373], the probability of bit error is

$$P_b(E) = \left( \frac{1}{m+1} \right) \left( \frac{1}{1 - P_r} \right) \left\{ (2^m(m+1) - 1) \sum_{k=0}^{t_{\max}} P_k 2^{m-1} + \sum_{k=0}^{t_{\max}} P_k 2^m \right\} \quad (9)$$

The probability of retransmission on a given transmission is the probability that the received vector falls into neither the correct decoding sphere (that surrounding the all-zero code word) nor any other. Thus

$$P_r = 1 - P_e - \sum_{e=0}^{t_{\max}} \binom{2^m}{e} p^e (1 - p)^{2^m-e}. \quad (10)$$

The throughput of the system shall be defined as the number of correct bits sent to the data sink per bit transmission interval. Assuming a selective-repeat retransmission protocol [8], the distinct packets (each containing $(m+1)$ data bits combined with $(2^m - m - 1)$ redundant bits) can expect to be transmitted $T_I$ times each, where

$$T_I = \frac{1}{1 - P_r}. \quad (11)$$

The subscript of $T_I$ is used to denote that a type-I hybrid-ARQ protocol is being used. The throughput of the system is then

$$\eta = \frac{(m+1)}{2^m} \left( \frac{1 - P_b(E)}{T_I} \right). \quad (12)$$

### 3.2 Performance examples

The following is a collection of computed performance curves for a type-I RM hybrid-ARQ protocol used in conjunction with a BPSK modem over an AWGN channel. The $R(1,4)$ code (length 16, dimension 5, and minimum distance 8) has been selected. The set of curves in Figure 1 compares the uncoded bit error rate to the decoded bit error rate provided by the type-I protocol for various
values of \( t_{\text{max}} \). The bit energy \( E_b \) for the three coded curves is the transmitted bit energy weighted by the code rate (5/16). The type-I protocol clearly provides a substantial improvement in reliability performance over the uncoded and FEC cases (the latter is slightly worse than the case \( t_{\text{max}} = 3 \)), particularly for the lower values of \( t_{\text{max}} \).

The impact of the retransmissions on system throughput is shown in Figure 2. The retransmission requests have a definite impact, particularly for the lower values of \( t_{\text{max}} \). However, even at the lower signal-to-noise ratios there is a reasonable amount of throughput and good reliability performance. The throughput curves do not tend to zero asymptotically as the SNR degrades because the probability of eventually receiving a correct bit is nonzero even when the raw channel bit error rate is 1/2.

Figure 1: Bit error rate curves for the type-I protocol using \( R(1,4) \).

If one wants to operate the proposed type-I protocol at very low signal to noise ratio, however, these two sets of curves indicate that performance will be marginal for some applications. In the next section a method is investigated for improving
4 The Reed-Muller type-II hybrid-ARQ protocol

Type-II hybrid-ARQ protocols provide the simplest example of code combining schemes, techniques by which multiple received packets are combined to create more reliable packets [9], [11], [12], [13]. In a type-II protocol two packets from a rate $R$ code are combined to create a single noise corrupted code word from a rate $R/2$ code in the hope that the additional error correction capability provided by the lower rate code will be sufficient to ensure the acceptance of the packet. The RM type-II protocol is based on the following theorem.
Theorem 1 Let \( c = (c_1, c_2, \ldots, c_n) \) be a binary \( n \)-tuple. The construction \((c|c)\) shall be defined as the \( 2n \)-tuple of the form \((c_1, c_2, \ldots, c_n, c_1, c_2, \ldots, c_n)\). The construction \((c|c)\) is a code word in \( R(1, m+1) \) if and only if \( c \) is a code word in \( R(1, m) \).

Proof:
Let the set \( A \) consist of all code words in \( R(1, m+1) \) associated with Boolean functions of the form \( f = a_0 1 + a_1 v_1 + a_2 v_2 + \cdots + a_m v_m + 0 \cdot v_{m+1} \), where the \( a_i \)'s are arbitrary binary coefficients.

The argument values used in the construction of the truth table are assumed to take on a natural ascending order, with \( v_{m+1} \) being in the most significant position. The elements \( \{v_1, v_2, \ldots, v_m\} \) will thus begin to repeat at the midpoint of the truth table. The code words in \( A \) are thus all of the form \((c|c)\) for some collection of binary \( 2^m \)-tuples \( \{c\} \).

Since the elements in \( \{c\} \) are the truth tables for the set of all Boolean functions in \( m \) variables, \((c|c) \in A \iff c \in R(1, m) \).

The elements \( \{1, v_1, v_2, \ldots, v_m\} \) are linearly independent \( \Rightarrow |A| = 2^{m+1} \).

Now let the set \( B \) consist of all code words in \( R(1, m+1) \) associated with Boolean functions of the form \( f = a_0 1 + a_1 v_1 + a_2 v_2 + \cdots + a_m v_m + 1 \cdot v_{m+1} \), where the \( a_i \)'s are arbitrary binary coefficients.

Since \( v_{m+1} \) takes the value 0 in the first half of the truth table and the value 1 in the second half, the code words in \( B \) must be of the form \((c|\bar{c})\), where \( \bar{c} \) is the one's complement of \( c \).

The elements \( \{1, v_1, v_2, \ldots, v_m\} \) are linearly independent \( \Rightarrow |B| = 2^{m+1} \).

\( \bar{c} \neq c \) for any \( c \Rightarrow A \cap B = \emptyset \).

\( \Rightarrow A \cap B = 2^{m+2} = |R(1, m+1)| \)
\( \Rightarrow (c|\bar{c}) \in R(1, m+1) \iff c \in R(1, m) \).

QED

The collection of code words \( \{(c|\bar{c}) \in R(1, m+1)\} \) forms a linear subcode of \( R(1, m+1) \) which shall be designated \( R'(1, m+1) \). The non-zero weight enumerators for \( R'(1, m+1) \) are

\[ A_0 = A_{2m+1} = 1, \quad A_{2m+1} = 2^{m+1} - 2. \]  

(13)

Reliability performance is improved by decoding code words of the form \((c|c)\) using decoders designed for \( R'(1, m+1) \) instead of \( R(1, m+1) \). Though the two
codes have the same minimum distance, \( R(1, m+1) \) has almost twice as many code words with weight \( 2^{m+1} \). In lattice theoretic terms one would say that \( R'(1, m+1) \) offers no better sphere packing performance, but has approximately half the kissing number of \( R(1, m+1) \). In a later section it is shown that a decoder for \( R'(1, m+1) \) can be easily obtained from the Green machine designed for \( R(1, m) \).

A RM type-II hybrid-ARQ protocol can now be defined in the following manner. It is assumed that the transmitted packets are encoded using \( R(1, m) \).

1. The first copy of the packet is transmitted. The receiver attempts to decode the packet using the \( R(1, m) \) type-I hybrid-ARQ decoder with maximum allowed error correction \( t_1 \). If no retransmission request is generated, an ACK is sent to the transmitter and the data is forwarded to the data sink. If a retransmission request is generated, the packet is saved in a buffer, a NACK is sent to the transmitter, and the process continues.

2. The second copy of the packet is transmitted. The receiver attempts to decode the second packet using the type-I \( R(1, m) \) decoder. If no retransmission request is generated, an ACK is sent to the transmitter and the data is forwarded to the data sink. If a retransmission request is generated, the process continues as follows.

3. The receiver concatenates both received copies of the packet and attempts to decode using the \( R'(1, m+1) \) decoder. If no retransmission request is generated (\( t_r \leq t_2 \)), the resulting code word is checked to see if it is of the form \((c|c)\). If so, an ACK is sent to the transmitter and the data is forwarded to the data sink. If a retransmission request is generated or the code word selected is not of the form \((c|c)\), a NACK is sent to the transmitter, all received packets are discarded, and the process starts over at step 1.

4.1 Reliability and throughput performance

The following generic graph describes the ways in which a data packet can proceed from packet initiation (\( PI \), the initial coding and transmission by the transmitter) to packet acceptance (\( PA \), the decoding of the packet without the generation of a retransmission request) in a type-II hybrid-ARQ protocol. The states \( D_i \) in the graph correspond to decoding operations on single packets (\( D_1 \)) and decoding operations on combined packets (\( D_2 \)). The graph is used to compute reliability.
performance $P(E)$ and the value of $T_{II}$, the expected number of times a copy of the packet must be transmitted before it is accepted in the type-II protocol.

Figure 3: Packet processing graph for type-II hybrid-ARQ protocols.

The transfer function for the graph in Figure 3 is

$$PA = \{ab + acd + acef + acegb + acegcd + \cdots\}PI$$
$$= \{a(b + cd + cef)[1 + ceg + (ceg)^2 + \cdots]\}\{PI$$
$$= \left\{ a\left[b + c(d + e + f)\right]\left(\frac{1}{1 - ceg}\right) \right\}PI$$  \hspace{1cm} (14)

Table 1 lists the labels used in the derivation of the throughput and reliability expressions. The superscripts on the probabilities of retransmission ($p_r^{(i)}$) and probabilities of decoder error on a given transmission ($p_e^{(i)}$) are used to reference the probability to the decoding of a single packet ($i = 1$) or the concatenation of two packets ($i = 2$).

The word error rate among accepted packets is readily obtained through substitution of the labels on the right-hand side of the chart into Equation (14). The following expression results.

$$P(E) = \left[ p_e^{(1)} + p_r^{(1)} \left(p_e^{(1)} + p_r^{(1)}p_e^{(2)}\right) \right] \left(\frac{1}{1 - p_r^{(1)}^2 p_r^{(2)}}\right)$$  \hspace{1cm} (15)
The probability of bit error is obtained by substituting the expressions developed below for the decoder error probabilities and retransmission probabilities used in Equation (15). The retransmission expression $p_r^{(1)}$ for the single packets is same as derived in Equation (10). The expression for probability of bit error for the decoding of the first packets (branches b and d in Table 1) is $p_b^{(1)}$ and is derived using Equations (7) and (9) as follows:

$$p_e^{(1)} = p_b^{(1)} = \frac{1}{m+1} \left\{ (2^m(m+1) - 1) \sum_{k=0}^{t_1} P_k^{2m-1} + \sum_{k=0}^{t_1} P_k^{2m} \right\},$$  

(16)

where $t_1$ is the maximum error correction for a single packet.

The calculation of the values for the branches associated with the decoding of the combined packet $p_e^{(2)}$ (and thus $p_b^{(2)}$) is more complex. The bit error rate of the combined packet is affected by the knowledge of the initial decoding operation of the component packets. Let

$$p_e^{(2)} = p_b^{(2)} = \frac{1}{m+1} \left\{ ((m+1)2^m - 1) \sum_{k=0}^{t_2} P_k^{2m} + \sum_{k=0}^{t_2} P_k^{2m+1} \right\},$$  

(17)

where

$$P_{k}^{ij} = \sum_{r=0}^{k} \binom{j}{k-r} \binom{2^m-j}{r} p'_{j-k+2r} (1 - p')^{2^m-j+k-2r},$$  

(18)

and $t_2$ is maximum error correction for the combined packet operation.

The BSC crossover probability $p'$ used in Equation (18) is not the same as that used in Equation (6). A modified BSC crossover probability is computed to
take into account the knowledge that there were sufficient numbers of errors in both packets to cause the generation of retransmission requests during the first two decoding attempts.

Let $p$ be the BSC crossover probability as defined by the channel noise level, transmitter power level, and binary modulation format. The crossover probability $p$ shall be referred to as the a priori BER in the received packets because it can be used to compute the expected number of errors in a received packet before decoding is attempted. The a posteriori BER $p'$ shall be the BER in packets that are known to have generated retransmission requests. The a posteriori BER $p'$ is computed in the following manner. First note that for a received packet $r$ of length $n$,

$$p = \frac{np}{p} = \frac{\text{Expected number of errors in } r}{\text{length of } r}. \quad (19)$$

The expected number of errors in $r$ can also be expressed as

$$\text{Expected number of errors in } r = np = \sum_{e=0}^{n} e \left[ \binom{n}{e} p^e (1-p)^{n-e} \right]. \quad (20)$$

If $r$ is known to have caused the generation of a retransmission request, some of the error patterns counted in the summation in Equation (20) are no longer valid possibilities. All error patterns that place the received packet within one of the decoding spheres must be deleted from the summation. The remaining probabilities are then normalized so that they still sum to unity (according to Baye's Law). Let $\Omega$ be defined as the expected number of errors in $r$ given that $r$ has caused the generation of a retransmission request. Using Equations (6), (7), and (20) and knowledge of the length and weight distribution of $R'(1,m+1)$, the following expression for $\Omega$ results:

$$\Omega = np - \sum_{e=0}^{n} e \binom{2^m}{e} p^e (1-p)^{2^m-e} - \sum_{k=0}^{t_1} 2^m - k \binom{2^m}{k} p^k (1-p)^{2^m-k} - (2^m+1) - 2) \sum_{k=0}^{t_1} \sum_{r=0}^{k} \left( \binom{2^m-1-k+2r}{k-r} \binom{2^m-1}{r} p^{2^m-1-k+2r} (1-p)^{2^m-1+k-2r} \right), \quad (21)$$

where $t_1$ is the maximum error correction in the first decoding operation.

The a posteriori BER $p'$ is then

$$p' = \frac{\Omega}{nP_r}, \quad (22)$$

13
where $P_r$ is the retransmission probability of Equation (10).

Figure 4: A comparison of the A Priori BER to the A Posteriori BER for received packets encoded with $R(1, 4)$.  

Figure 4 shows how the a posteriori BER differs from the a priori BER for the $R(1, 4)$ code used with a BPSK modem over an AWGN channel. Clearly a substantial overestimation of the performance of the type-II protocol is made unless the knowledge that a retransmission request has been generated by a packet is taken into account. The a posteriori BER is, as expected, higher than the a priori BER as the combined packet decoding operation uses packets that are known to contain some number of errors.

The retransmission request probability $p_r^{(2)}$ for the combined packet, also a function of the a posteriori BER, is derived as follows:

$$p_r^{(2)} = (2^{m+1} - 2) \sum_{k=0}^{t_2} P_k^{2^m} + \sum_{k=0}^{t_2} P_k^{2^{m+1}}$$
where $P'_i$ is defined in Equation (18). Using the modified or \textit{a posteriori} BER the values for $p_e^{(2)}$ (and thus $p_b^{(2)}$) and $p_r^{(2)}$ are easily found and the probability of bit error for the type-II system is determined to be

$$P_b(E) = \left[ p_b^{(1)} + p_r^{(1)} \left( p_b^{(1)} + p_r^{(2)} \right) \right] \left( \frac{1}{1 - p_r^{(1)} T^2} \right).$$

The expected number of transmissions required for each data packet in the type-II system ($T_{II}$) is derived through the use of generating function techniques. Using the labels in the middle column of Table 1, the various paths through the graph in Figure 3 are assigned weights of the form $PT^x$, where $P$ is the probability that the path will be taken and $x$ is the number of packet transmissions made before the path terminates in packet acceptance. The following generating function is obtained through substitution from Table 1:

$$G(T) = T \left[ 1 - p_r^{(1)} + p_r^{(1)} \cdot T \left( 1 - p_r^{(1)} + p_r^{(1)} \left( 1 - p_r^{(2)} \right) \right) \right] \left( \frac{1}{1 - p_r^{(1)} T^2} \right).$$

The expected value of $T_{II}$ is obtained by taking the partial derivative of $G(T)$ with respect to $T$ and setting $T$ equal to 1. The throughput of the system is thus

$$\eta = m + 1 \cdot \left[ 1 - P_b(E) \right] \cdot \left( \frac{\partial}{\partial T} G(T) \right)_{T=1}^{-1}$$

$$= m + 1 \cdot \left[ 1 - P_b(E) \right] \cdot \left( \frac{1 + p_r^{(1)}}{1 - p_r^{(2)}} \right).$$

4.2 The modified Green machine for type-II hybrid-ARQ protocols

The Green machine provides a fast and efficient method for decoding first-order RM codes [1]. In this section a method is shown for modifying the Green machine for use with the RM type-II hybrid-ARQ protocol discussed earlier.
The primary function of the Green machine is the computation of the Hadamard transform of the received packets. This is efficiently accomplished by taking advantage of the Fast Hadamard Transform Theorem [1].

Let \( A \) be an \((m \times m)\) matrix and \( B \) an \((n \times n)\) matrix with elements \( \{a_{ij}\} \) and \( \{b_{ij}\} \) respectively. The Kronecker product \((A \otimes B)\) is the \((mn \times mn)\) matrix obtained by replacing every entry \( a_{ij} \) in \( A \) by the matrix \( a_{ij} B \).

Example:

\[
\begin{bmatrix}
a & b \\
c & d
\end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix}
a & b & b & b \\
a & b & b & b \\
c & c & d & d \\
c & c & d & d
\end{bmatrix}
\]

Theorem 2 (The Fast Hadamard Transform Theorem) \( H_{2^m} = M_{2^m}^{(1)} M_{2^m}^{(2)} \cdots M_{2^m}^{(m)} \), where \( M_{2^m}^{(i)} = I_{2^{m-i}} \otimes H_2 \otimes I_{2^{i-1}}, \ 1 \leq i \leq m, \ I_n \) is an \((n \times n)\) identity matrix, and

\[
H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

Proof: (see [1, p. 422])

The Hadamard transform is computed through the matrix multiplication \( F H_{2^m} \). The Green machine performs this operation through a series of much simpler matrix multiplications of the form \( F \cdot M_{2^m}^{(1)} M_{2^m}^{(2)} \cdots M_{2^m}^{(m)} \). The decoder for \( R(1,m) \) has \( m \) stages, the \( j^{th} \) of which is used to perform matrix multiplication by \( M_{2^m}^{(j)} \). Let \( F = (F_1, F_2, \cdots, F_n) \) be the input to the decoder. The matrix product computations in the \( R(1,m) \) decoding operation are noted below along with the associated stage of the Green machine.

- **1st Stage** \( \leftrightarrow \) \( F \cdot M_{2^m}^{(1)} = P^{(1)} \)
- **2nd Stage** \( \leftrightarrow \) \( P^{(1)} \cdot M_{2^m}^{(2)} = P^{(2)} \)
- \( \vdots \)
- **\( m^{th} \) Stage** \( \leftrightarrow \) \( P^{(m-1)} \cdot M_{2^m}^{(m)} = P^{(m)} = \hat{F} \)

The \( m \) stages of the \( R(1,m) \) decoder are closely related to the first \( m \) stages of the \( R(1,m+1) \) decoder by the following Lemma.
Lemma: $M_{2m+1}^{(i)} = \left[ \begin{array}{c} M_{2m}^{(i)} \\ M_{2m}^{(i)} \end{array} \right]$ for $i = 1, \cdots, m$. 

Proof:

$$
M_{2m+1}^{(i)} = I_{2m+1-i} \otimes H_2 \otimes I_{2i-1} \\
= (I_2 \otimes I_{2m-i}) \otimes H_2 \otimes I_{2i-1} \\
= I_2 \otimes (I_{2m-i} \otimes H_2 \otimes I_{2i-1}) \\
= I_2 \otimes M_{2m}^{(i)}
$$

QED

Let $F' = (F_1, F_2, \cdots, F_{2n})$ be the input to the $R(1, m+1)$ decoder. By separating $F'$ into two $n$-bit segments $F'_1 = (F_1, F_2, \cdots, F_n)$ and $F'_2 = (F_{n+1}, F_{n+2}, \cdots, F_{2n})$, the first $m$ stages of decoding can be performed by an $R(1, m)$ decoder through the exploitation of the Lemma.

$$
1^{st} \text{ Stage } \leftrightarrow F' \cdot \left[ \begin{array}{c} M_{2m}^{(1)} \\ M_{2m}^{(1)} \end{array} \right] = (F'_1 \cdot M_{2m}^{(1)} | F'_2 \cdot M_{2m}^{(1)}) = (P_1^{(1)} | P_2^{(1)}) \\
2^{nd} \text{ Stage } \leftrightarrow (P_1^{(1)} | P_2^{(1)}) \cdot M_{2m+1}^{(2)} = (P_1^{(1)} \cdot M_{2m}^{(2)} | P_2^{(1)} \cdot M_{2m}^{(2)}) = (P_1^{(2)} | P_2^{(2)}) \\
\vdots \\
m^{th} \text{ Stage } \leftrightarrow (P_1^{(m-1)} | P_2^{(m-1)}) \cdot M_{2m+1}^{(m)} = (P_1^{(m-1)} \cdot M_{2m}^{(m)} | P_1^{(m-1)} \cdot M_{2m}^{(m)}) \\
= (P_1^{(m)} | P_2^{(m)}) \\
(m+1)^{st} \text{ Stage } \leftrightarrow (P_1^{(m)} | P_2^{(m)}) \cdot M_{2m+1}^{(m+1)} = (P_1^{(m+1)} | P_2^{(m+1)}) = \hat{F}
$$

Figure 5 is a block diagram for a type-II decoder for packets encoded using $R(1, 3)$. The first three stages are used to decode the first received copy of each transmitted packet. The third stage is followed by a thresholding device that decides, based on whether the number of errors corrected in the packet exceeds $t_1$, to request a retransmission, or to forward the corresponding data to the data sink. If a retransmission request is generated, the results of the first decoding operation are stored in the left-hand register immediately following the third stage.

The above process is then repeated with the second received copy of the packet. If the number of errors corrected once again exceeds $t_1$, then the output of the
Figure 5: Modified Reed-Muller type-II decoder for the RM (1, 3) code.
third stage is concatenated with the corresponding output derived from the first copy of the packet. The resulting 16-bit vector \( P_1^{(3)} | P_2^{(3)} \) is then sent to the fourth stage of the modified Green machine. The output of the fourth stage designates a code word in \( R(1,4) \). If the number of errors corrected exceeds \( t_2 \) or if the designated code word is not of the form \((c|c)\) (i.e., if the code word is not in \( R'(1,4) \)) then a retransmission request is sent back to the transmitter. Otherwise the transmission is acknowledged and the data forwarded to the data sink. If the concatenated vector causes the generation of a retransmission request, the registers in the decoder are emptied. This is because any packet that causes a retransmission request in both decoding operations is in all likelihood too noisy to be of use in any further code combining efforts. (This is a necessary concession to the weakness of long RM codes.)

4.3 Performance examples

![Throughput performance for type-I and type-II protocols using R(1,4).](image)

Figure 6: Throughput performance for type-I and type-II protocols using \( R(1,4) \).
Figures 6 and 7 compare the performance of the type-I and type-II hybrid-ARQ protocols based on the first-order RM code of length 16. Figure 6 shows that the type-II protocol offers significantly better throughput performance than the type-I protocol at low SNR’s. Figure 7 shows that in this same operating region, the reliability of the two systems is virtually identical, though at higher SNR’s the type-I protocol begins to outperform the type-II protocol.

In Figures 8 and 9 one of the type-II protocols is compared to a type-II protocol based on a punctured (16,8) Reed-Solomon (RS) code [9]. Since RS codes are maximum distance separable, the RS type-II protocol provides the greatest possible incremental redundancy for a fixed-length retransmission. Figure 8 shows that the BER performance of the RM protocol is within an order of magnitude of the RS protocol at low and moderately low SNR’s. Although the BER performance is comparable, the Reed-Solomon code offers significantly better throughput performance, as is demonstrated in Figure 9. The improved performance of the RS protocol is achieved at the expense of a substantial increase in the complexity of
the decoder. The RM system may thus be preferable in high data rate and/or low cost applications.

\[
\begin{align*}
\text{Eb/No} \\
-10 -9 -8 -7 -6 -5 -4 -3 -2 -1 1 2 3 4 5 6 7 8 9 10
\end{align*}
\]

Figure 8: Reliability performance for type-II \( R(1,4) \) protocol and the punctured \( (16,8) \) Reed-Solomon code.

5 Conclusions

It has been shown that the Green machine decoder can be modified to allow for the use of first-order RM codes in both type-I and type-II hybrid-ARQ protocols. The type-I protocols offer substantially better reliability performance than their FEC counterparts, though at the expense of a reduction in throughput at low SNR’s. The type-II protocols use a simple code combining decoder that concatenates pairs of received packets to create lower rate code words, improving the chances of successful error correction. The type-II protocols offer similar reliability performance to that provided by the type-I protocols, but offer much better throughput performance at low SNR’s. Both the type-I and type-II protocols are
Figure 9: Throughput for type-II $R(1, 4)$ protocol and the punctured (16, 8) Reed-Solomon code.

obtained though minor modifications of the Green machine that do not appreciably increase the complexity or decrease the speed of the decoder.

References


