PERFORMANCE EVALUATION OF ADAPTIVE HYBRID ARQ SYSTEMS

by

Sattar E. Bakhtiyari
B. A. Sc., The University of British Columbia, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
DEPARTMENT OF ELECTRICAL ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

October 1993

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature)

Department of Electrical Engineering

The University of British Columbia
Vancouver, Canada

Date Oct. 28 / 93

Abstract

In the analysis of ARQ schemes, it is typically assumed that the feedback channel is noiseless and perfect synchronization can be achieved. These assumptions may not be valid for the situations where the channel error rate is high especially with memory ARQ schemes. In such cases, a more realistic evaluation of the performance is required to provide accurate results for the schemes in question. A general investigation of ARQ schemes with and without memory, taking into account the sensitivity of frame header and return channel errors, is presented and the resulting throughput expressions are obtained. Knowing the influence of these parameters on the system performance, we propose efficient techniques to reduce their effects on the performance degradation. These techniques range from employing more powerful error correction codes to transmitting a few copies of the same sequence for the header and acknowledgment message consecutively and combining the noisy sequences at the receiver.

Even though ARQ schemes offer appreciable throughput improvement in AWGN, their performance suffers greatly in the Rayleigh fading environment. In such a scenario, the optimum solution is to adapt the rate of error correction code to the prevailing channel conditions. An adaptive Type-I Hybrid ARQ (HARQ) scheme, in which the transmitter selects the code rate for each message based on the assessed channel conditions, is proposed and investigated. A model for evaluating the throughput of the scheme is presented, and it is illustrated that an accurate assessment of the channel conditions is crucial in the achievement of high throughput. Moreover, an adaptive Type-II HARQ with Complementary Punctured Convolutional Codes (CPC) is presented. This protocol stores the corrupted copies of a packet encoded with CPC codes at the receiver and combines them to improve the reliability of the system and reduce the number of unnecessary retransmissions.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I am indebted to my supervisor Dr. Samir Kallel for his support and continual guidance. Without his constant encouragement and valuable suggestions, the completion of this thesis would not have been possible.

I would like to express my special appreciation to my parents for their moral support and continuing affection throughout my studies.

I also would like to acknowledge the assistance provided by the B.C. Science Council.

Chapter 1 Introduction

Automatic Repeat Request (ARQ) is a technique for ensuring reliable data communication over noisy and fluctuating channels. Data is sent over a data link by packets (frames) to which parity bits are added for error detection purposes. If the received packet contains errors, the receiver sends the transmitter a negative acknowledgment message requesting the retransmission of the same packet. ARQ is useful as long as the channel bit error rate is low. However, in highly degraded channel, many retransmissions of the same packet may be required resulting in low throughput.

An ARQ technique which employs error correction codes is referred to as Hybrid ARQ (HARQ). In HARQ, since some error patterns in the noisy packet may be corrected, a packet will require fewer retransmissions than the case of Basic ARQ. Hybrid ARQ schemes are classified as Type-I and Type-II HARQ.

Type-I HARQ scheme employs one code for error detection and another for error correction purposes. When a received packet is detected in error, the receiver first attempts to correct the errors. If the number of errors is within the error-correcting capability of the code, the errors are corrected. Otherwise, the packet is rejected and a retransmission is requested. The drawback of this scheme is that the code rate is fixed. As a result, all parity bits for error correction are transmitted even if they are not needed. To remedy this problem and achieve optimum performance, the rate of error-correction codes should be matched to the channel conditions. This idea forms the basis of Type-II HARQ [1]. In this scheme, the transmitter first sends the data encoded with error–detecting bits. If this transmission is

unsuccessful, the error-correcting bits are transmitted. Thus, we avoid wasting the channel capacity at low channel bit error rates and still have the power of an error correcting code available when it is needed.

Other Type-II HARQ schemes, that are aimed at channels with varying conditions, have been proposed and analyzed in [2–4]. In these schemes, the packet is encoded by an error-correcting code and sent to the receiver. If the decoding is unsuccessful, the packet is retransmitted and the two versions of the packet are combined at the receiver to obtain an effective lower rate code. If decoding attempt fails, additional retransmissions are requested. The receiver stores all the copies of the received packet and attempts to decode them under all possible combinations.

Type-II HARQ protocols fall into the category of ARQ schemes with memory (MARQ) which provide memory at the receiver for the purpose of correcting errors. These schemes make use of corrupted copies of a packet to reduce the average number of retransmissions. Several variations of these MARQ systems have been proposed and investigated in [2], [5], [6], [7]. Some of these techniques such as reliability updating process or multiple copy combining lead to correct recovery of the message without the exploitation of any explicit error correction codes.

All ARQ schemes are typically analyzed without taking into consideration the effects of feedback channel errors and frame header sensitivity. It is usually assumed that the feedback channel is noiseless and perfect synchronization can be achieved. Although these assumptions may be valid for channels with low error rates, with increasing error rates more accurate analysis is desirable for situations such as memory ARQ systems. Memory ARQ schemes

allowing the exploitation of repetition redundancy are very sensitive to these parameters. In other words, the advantage offered by these strategies may not be completely achievable in the cases where frequent packet losses occur due to frame header failure. We, therefore, investigate the performance of ARQ systems while the effects of feedback channel errors and frame header sensitivity are taken into account. An efficient technique for calculating the expected number of transmission for a packet is provided. The analysis makes use of state diagrams which are constructed according to the ARQ transmission strategy. Efficient techniques are also proposed to reduce the influence of the indicated parameters on the system performance to a minimum level.

As mentioned earlier, fixing the rate of the error-correcting code may lead to unsatisfactory throughput in fluctuating channels. If the channel is quiet, the redundancy in the system may be more than the optimum value. On the other hand, if the channel is fairly noisy, more errors are expected to occur than those which can be handled by the capability of the error correcting code. The optimum solution is to match the code for error correction to the prevailing channel conditions. An adaptive Type-I HARQ is proposed and analyzed in the time-varying channels. During poor channel conditions, a low-rate code is used to minimize the number of retransmissions, whereas with improvement in the channel conditions high-rate codes are employed to reduce the transmission of unnecessary overhead. An efficient algorithm similar to that in [8] is proposed to facilitate the estimation of the channel conditions for the purpose of code adaptation. The theoretical analysis for the throughput of the adaptive scheme is conducted. Numerical and simulation results using Convolutional Codes are presented.

In the adaptive Type-I HARQ scheme, the packets detected in error are discarded. An Adaptive Type-II HARQ with Complementary Punctured Convolutional Codes (CPC) is proposed to take advantage of these erroneous packets being discarded at the receiver. This protocol functions in the same manner as the preceding adaptive scheme except it stores the encoded packets with CPC codes from different receptions and combines the minimum number of available equivalent codes to overcome the channel noise and recover the message.

The outline of the thesis is as follows:

Chapter 2 is a brief review of the basic concepts essential in understanding the ARQ systems. In chapter 3, the throughput expressions for the ARQ schemes are developed and the effects of frame header and ACK sensitivity on the system performance are evaluated. Chapter 4 proposes an adaptive Type-I HARQ and various elements needed for its design. A probabilistic model is employed to ascertain the throughput of the proposed algorithm. The effects of interleaving/deinterleaving operations are also presented in this chapter. The second section of this chapter is dedicated to the description and evaluation of an adaptive Type-II HARQ scheme. Simulations have been run to show the effectiveness of the proposed schemes.

The contributions of this thesis to digital communication theory are as follows:

1. Derivation of throughput expressions for ARQ schemes with and without memory under the assumptions of noisy feedback channel and with frame header sensitivity.
2. Design and investigation of an adaptive Type-I and Type-II HARQ schemes for time-varying channels.

- Design of an efficient channel estimation algorithm for the purpose of code adaptation.
- Derivation of the throughput expression for the adaptive Type-I scheme with the assumption of noise-free return channel.
- Simulation of the adaptive Type-I scheme taking into account various parameters such as speed, observation intervals, and adaptation thresholds.
- Simulation and evaluation of the performance of the adaptive Type-II HARQ scheme.

Chapter 2 Preliminaries

This chapter provides a broad introduction to ARQ Systems and discusses the underlying concepts necessary for understanding the thesis. The fundamentals of Convolutional Codes, Viterbi Decoding, and Punctured Convolutional Codes are presented in section 1. Section 2 gives a brief description of the fading channel and employed modulation scheme. Various types of ARQ schemes with and without memory are described in section 3, followed by examination of frame structure and its components in section 4. Finally, section 5 investigates the significance of frame synchronization for reliable data communications.

2.1 Convolutional Codes

Convolutional Codes were first introduced by Elias [9]. These codes with hard and soft-decision Viterbi decoding have found applications in many Space and Satellite Communications [1]. The availability of efficient decoding strategies for these codes has also incited the development of new classes of codes such as Punctured Convolutional Codes.

2.1.1 Encoding of Convolutional Codes

An (n, k, m) Convolutional code is constructed with k-input, n-output linear sequential circuit with memory m. The memory m corresponds to the number of stages in the shift register, and it indicates the dependency of the n current output bits on the current and m previous inputs. The rate of the code, indicating the amount of redundancy in a coded sequence, is defined as the ratio k/n bits/coded symbol. These codes are most easily described by an example. Figure 2.1 shows a shift register circuit which generates a rate

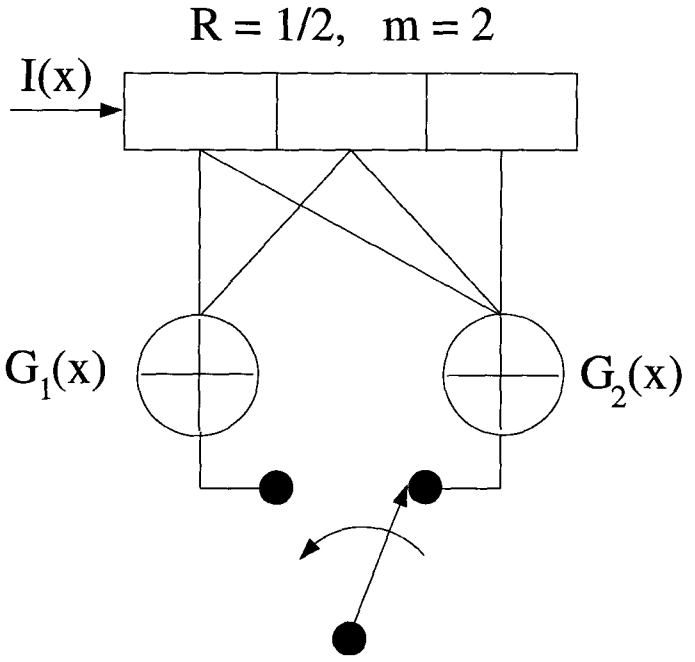

Figure 2.1 Block Diagram of a Convolutional Encoder with Rate 1/2

1/2 Convolutional Code. As the input bits are shifted into the left-most stage, the bits in the register are shifted one position to the right. The multiplexer samples the output of each modulo-2 adder. In this example, for each input bit, two output symbols are generated. We can see that a particular input bit influences the output during its own interval as well as the next two input bit intervals. Two ways to represent the encoder are with state and trellis diagrams which are shown in Figure 2.2. The states , shown by a=00, b=10, c=01, and d=01, represent the possible contents of the right-most m stages of the register, and the paths between the states represent the output branch words resulting from such state transitions. There are two transitions emanating from each state, corresponding to the possible input bits. Next to each path between states is written the output branch word associated with the state transitions. A solid line denotes a path associated with an input bit, zero, and a dashed line

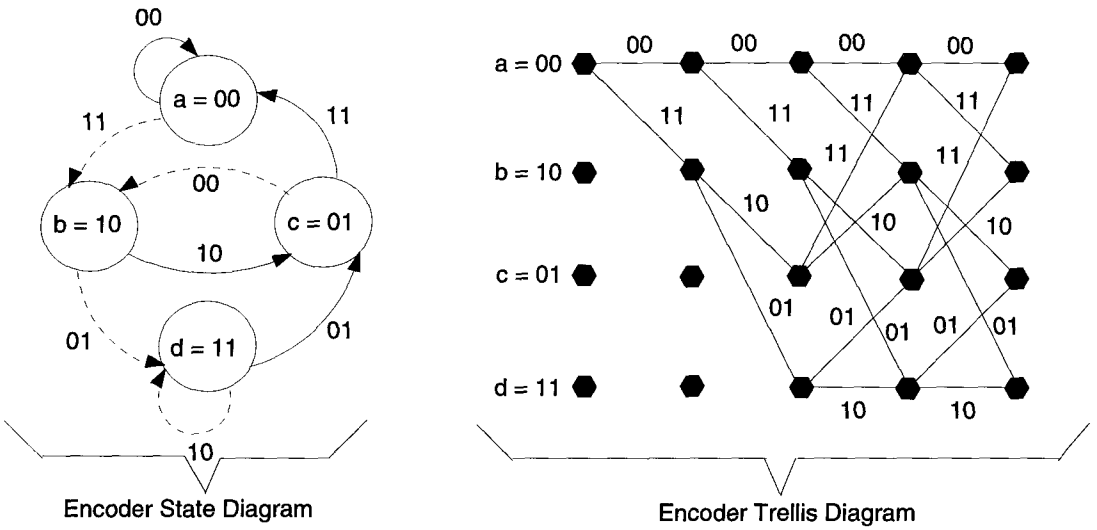

Figure 2.2 Encoder State and Trellis Diagrams

represents a path associated with an input, one. The trellis in this example assumes a fixed periodic structure after trellis depth 3 is reached. After this point, each of the states can be entered from either of two preceding states.

The choice of the connections between the adders and stages of the encoder is important. In other words, different connections result in different output codes. The encoder connections can be represented by generator polynomials, one for each of the n modulo-2 adders. The coefficient of each term in the m-degree polynomial is either 1 or 0, depending on whether a connection exists or does not exist between the shift register and the modulo-2 adder.

2.1.2 Viterbi Decoding

The Viterbi algorithm was discovered by Viterbi [10] in 1967. The algorithm takes advantage of the special structure in the code trellis. Each path in the trellis corresponds to a possible transmitted sequence. The algorithm calculates the measure of similarity (Maximum

Likelihood Function or metric) between the received sequence and all the possible paths in the trellis and selects the path which has the best resemblance to the received sequence. This path is called the surviving path.

In summary, given a received sequence, the Viterbi decoder calculates the path metric for every path through the trellis. The decoder output is the path which has the largest path metric. Estimating the trellis path is equivalent to estimating the transmitted information sequence because there is a one-to-one correspondence between the trellis path and the information sequence.

Long input sequences lead to an increase in the complexity of the algorithm. In practice, the decoding decision for the i-th bit is based on the best partial path at time (i + d_τ) where d_τ is called the truncation path length. It is shown [1] that a minimum truncation length of about 4 to 5 times the code memory is sufficient to ensure that the error probability due to truncation is negligible.

The decoding error, *P(E)*, and bit error probabilities, *P(B)*, for Convolutional Codes with Viterbi decoding are upper bounded [1] by

$$\begin{aligned} P(E) &\leq \sum_{d=d_{free}}^{\infty} a_d P_d \\ P(B) &\leq \frac{1}{k} \sum_{d=d_{free}}^{\infty} c_d P_d \end{aligned} \tag{2.1}$$

In these expressions, d_{free} is the free distance of the code and a_d is the number of incorrect paths of Hamming weight d that diverge from the correct path and merge with it sometimes later. As for c_d, it is the total number of bit errors on all the paths having a Hamming weight d. The Hamming weight (or simply weight) is defined as the number of non–zero

components in a code sequence. P_d is the probability that a wrong path at distance d is selected, and it depends on the type of modulation and channel parameters used. For Non-Coherent Frequency Shift Keying (NCFSK) modulation and an ideally interleaved Rayleigh fading channel with AWGN, P_d is given [11] by

$$P_d = p^d \sum_{j=0}^{d-1} \binom{d+j-1}{j} (1-p)^j, \tag{2.2}$$

where $p = \frac{1}{2+\gamma_0}$. with γ_0 being the average signal-to-noise ratio.

2.1.3 Punctured , Repetition, and Rate-Compatible Convolutional Codes

Punctured Convolutional Codes were first introduced by Cain, Clark, and Geist [12], and they are employed to enable adaptive encoding/decoding without modifying the basic structure of the encoder and the decoder. To explain the construction of these codes, we only consider Punctured and Repetition Codes obtained from original rate 1/2 codes. A high rate (V-1)/V Punctured Convolutional Code can be obtained from a rate 1/2 code by deleting (V-2) bits from every 2(V-1) coded bits corresponding to the encoding of (V-1) information bits according to a well selected perforation matrix. The perforation matrix yielding a rate (V-1)/V code has two rows and (V-1) columns. The elements of each row correspond to one of the two encoded bits at the output of the rate 1/2 encoder, and those of each column are associated to one encoding cycle. The elements of a perforation matrix are only zeros and ones, corresponding to deleting or keeping the coded bit at the output of the original rate 1/2 encoder. For example, the perforation matrix P_0 of a rate 7/8 Punctured Convolutional Code of memory m=6, obtained from a rate 1/2 is given [13] by

$$P_0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

A low-rate (V-1)/(2(V-1) + l), $l \geq 1$, Repetition Code can be obtained from a rate 1/2 code by repeating l bits among every 2(V-1) coded bits which result from the encoding of each group of (V-1) information bits [14]. The l bits being repeated are determined by a well-selected repetition matrix which has two rows and (V-1) columns. In contrast to the perforation matrix, each element of a repetition matrix is greater than or equal to one and indicates the number of repetitions of the corresponding coded bit. For example, the repetition matrix, Q, of a rate 7/17 Repetition Convolutional Code of memory m = 6, obtained form a rate 1/2, is given [14] by

$$Q = \begin{bmatrix} 2 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 2 & 1 & 1 \end{bmatrix}. \tag{2.4}$$

A family of Punctured and Repetition Codes can be obtained from the same original rate 1/2 code. Decoding of these codes can be performed using essentially the same decoder structure of the original rate 1/2 code, with only some minor modifications [13]. Using the rate 1/2 trellis structure for the decoding, a zero metric is assigned for those punctured bits; and for those repeated bits, a metric corresponding to each additional received bit is added.

Moreover, a constraint of rate compatibility [15] can be incorporated to the procedure of generating a family of Punctured and Repetition Codes. The rate compatibility restriction implies that all code bits of a given code of the family are used by all lower rate codes. A simple method for the construction of Rate Compatible Convolutional Codes (RCC) is presented in [14]. The RCC codes in [14] are derived from a known high rate (V-1)/V Punctured Convolutional Code which is obtained from a rate 1/2 code as follows. Starting

with the rate (V-1)/V code, Rate Compatible Codes of lower rates are obtained by simply adding sequentially, a few bits at a time, the (V-2) bits that were initially deleted to form the rate (V-1)/V code. When all (V-2) coded bits are used, we get a code of rate (V-1)/2(V-1) which is the original rate 1/2 code. Form this rate (V-1)/2(V-1) code, Rate Compatible Codes of lower rates are obtained by sequentially repeating, a few bits at a time, the 2(V-1) coded bits, without any limit. The rate (V-1)/(V-1) = 1 code can also be added to the family, provided it is invertible [16].

The advantage of RCC codes is that from a given sequence encoded with any code of the family, additional coded bits can be properly inserted in that sequence, yielding a lower rate encoded sequence. These Codes are especially useful in some rate-adaptive ARQ/FEC applications since only the incremental redundancy bits need to be transmitted as the coding rate is decreased.

2.1.4 Complementary Punctured Convolutional Codes (CPC)

The concept of equivalent codes must be understood first before we describe how these codes are constructed. Let C_1 and C_2 represent two Punctured Convolutional Codes of the same rate. If the code C_1 can be obtained from C_2 by shifting b columns of the Perforation matrix of C_2 or vice versa, then these two codes are said to be equivalent. Being equivalent, these codes have the same distance spectra. For example, the two rate 3/4 codes of perforation matrices

$$P_0 = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \quad P_1 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{2.5}$$

are equivalent.

In general, let C_i, i = 1, 2, ... t, be t equivalent codes of the same rate b/V all derived from an original code with rate $1/V_0$. Let P_i denote the perforation matrix of code C_i, then the perforation matrix P is defined as

$$P = \sum_{i=1}^{t} P_i. \tag{2.6}$$

The equivalent codes C_i, i = 1, ... , t are considered to be complementary if any element of the matrix P is equal or greater than 1 [4]. The resulting matrix P for the above example is

$$P = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.7}$$

which satisfies the stipulated condition for Complementary Punctured Convolutional Codes.

2.2 Fading Channel Model and NCFSK Modulation

A well-known fading model by Jakes [17] is adopted in this study. This model produces a random function resembling the strength of a multi-path fading signal. It is shown [17] that the envelope of the faded signal is Rayleigh distributed with probability density function

$$f(\alpha) = \frac{\alpha}{b_0} exp\left(-\frac{\alpha^2}{2b_0}\right), \qquad \alpha \geq 0 \tag{2.8}$$

where $b_0 = \frac{E_0^2}{2}$ with E_0 being defined as the energy of the transmitted signal. Throughout this thesis, we assume that b_0 = 0.50 so that E_0^2 = 1. Then, the expression (2.8) simplifies to

$$f(\alpha) = 2\alpha exp(-\alpha^2), \qquad \alpha \geq 0 \tag{2.9}$$

Therefore, the effects of the fading on the amplitude of the transmitted signal is the same as multiplying the base-band signal by the time varying fading amplitude α(t).

Non-coherent detections are employed in digital communications when the effects of the channel on the transmitted signal are unknown. In this thesis, we employ Frequency Shift Keying (FSK) Signalling with incoherent demodulation whose error probability is given by $p = \frac{1}{2+\gamma_0}$ [18], where γ_0 is the average signal-to-noise ratio defined as

$$\gamma_0 = SNR = E[\alpha^2]\frac{E_s}{N_0}. \tag{2.10}$$

N_0 is the single-sided power spectral density of Additive White Noise, $E[\alpha^2]$ is simply the average value of α^2, and E_s is the energy per symbol. The numerical and simulated BER curves of FSK signalling with incoherent detection are shown in Figure 2.3. It can be observed that the simulated results are in excellent agreement with the theoretical ones.

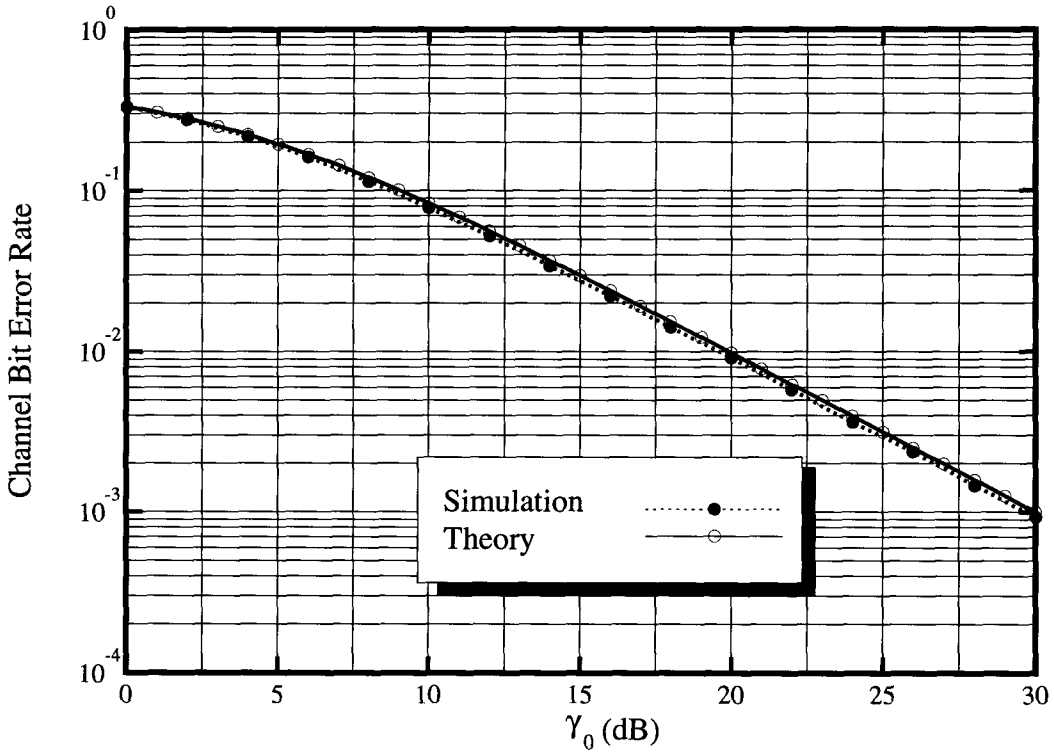

Figure 2.3 The Bit Error Rate Performance of FSK with Incoherent Demodulation

2.2.1 Assumptions for the Fading Channel

The following assumptions are made in the simulations of the fading channel for the ARQ schemes throughout the thesis.

1. Unless otherwise stated, the carrier frequency f_c has been chosen as 900 MHz, and the Baud rate as 9600 symbols/s. Therefore,

$$\begin{aligned} B_d &= \frac{v}{c} f_c = 0.833v \\ B_d T &= 8.69 \times 10^{-5} v \end{aligned} \tag{2.11}$$

   where B_d is the maximum Doppler shift, v is the speed of vehicle, and c is the propagation speed of light.

2. It is assumed that the fading is sufficiently slow so that it can be considered as constant at least during one symbol interval.

3. FSK with non-coherent demodulation is assumed.

2.3 Conventional and Hybrid ARQ Strategies

The ARQ schemes for controlling transmission errors in data transmission systems can be classified into two categories: ARQ schemes with memory and without memory. In ARQ schemes without memory, data packets detected in error after decoding are discarded, even though they may contain useful information. In contrast, ARQ schemes with memory use the simple idea of utilizing erroneously received packet to facilitate the recovery of the transmitted message. All ARQ schemes are used in conjunction with a retransmission protocol.

The three main retransmission protocols are Stop-and Wait (SW), Go-back-N (GBN), and Selective Repeat (SR). The SW protocol, despite its implementation simplicity, is inefficient

due to the idle time spent waiting for the receiver to acknowledge each transmission. The GBN scheme, in which the transmitter continuously sends packets one after another, is more efficient. However, the performance of the GBN protocol is not satisfactory at high channel error rates or large round-trip delays. The SR ARQ protocol, in which the transmitter repeats only the packets detected in error, ranks as the most efficient one in terms of throughput performance. However, the price paid is the increased system complexity [1].

2.3.1 ARQ Schemes without Memory

In these schemes, packets containing errors are discarded. These strategies yield satisfactory performance when the channel error rate is low. However, their performance as measured by throughput deteriorates with increasing channel error rates.

2.3.1.1 Type-0 ARQ Strategy

Each packet of length n consisting of k information and n_p CRC bits for error detection is transmitted. At the receiver, if the decoded packet contains errors, it is discarded and a request for retransmission is made. The main drawback of this ARQ technique is the necessity of retransmitting a packet many times in the case of low signal-to-noise ratios. As a result, the number of retransmissions needed before receiving a packet correctly may be very high culminating in the reduction of the throughput to an unacceptable extent.

2.3.1.2 Type-I Hybrid ARQ Strategy

An error control technique providing improvement over the Type-0 ARQ Strategy is referred to as Type-I Hybrid ARQ. This scheme employs both error correction and detection codes to combat the channel degradation. At the receiver, the decoder first attempts to

correct as many errors as possible in the received packet. Then, the decoded information bits are tested for error detection. If no errors are detected, the message is delivered to the user. Otherwise, the receiver requests the retransmission of the same packet. Several type-I Hybrid ARQ schemes using either Block or Convolutional Codes have been proposed and analyzed [19–24]. Type-I ARQ schemes are best suited for communication systems, in which the channel conditions are stationary and when a fairly constant level of noise is anticipated. However, for channels having non-stationary behavior, the throughput of the system falls rapidly.

2.3.2 ARQ Schemes with Memory (MARQ)

The basic idea is that the packet detected in error should not be discarded because it contains useful information about the transmitted packet. Instead, the packet is retained at the receiver to make use of the information contained in it. If the retransmission of the packet is also in error, the collective information present in the first packet and the retransmission can be used to correct certain errors in the two copies of the packet. In case error recovery is still not possible, additional retransmissions can be used until enough redundancy is present to allow correction. If such measures are incorporated in the simple and hybrid ARQ systems, the average number of retransmission will decrease substantially.

2.3.2.1 Type-0/I MARQ Schemes

These schemes make use of the several transmissions of a single packet even if they contain errors to try to reduce the average number of retransmissions.

In the Type-0 MARQ, the message of length k-bit is encoded into an n-bit packet. The packet is transmitted over the forward channel. At the receiver, the packet is checked to see whether the message can be recovered. If it can't, the packet is stored in a receiver buffer, and a retransmission is requested. If the retransmission of the packet contains errors, the first packet is combined with retransmitted one as suggested in [5–7]. When additional retransmissions are required, the same procedure for combining the successive copies of the packet is repeated. In [7], a reliability vector is associated to each received packet, and it is updated every time a new repetition of the same packet is received. The errors are, then, corrected by the reliability updating process without any explicit error-correction procedure.

The Type-I MARQ operates in the same manner as the Type-0 MARQ except for the exploitation of error correction codes. Therefore, the message of length k is encoded with a block code for error detection followed by an error correction code. The resulting packet is transmitted over the channel. At the receiver, decoding is performed on the received packet. If the decoded message contains error, the packet is buffered at the receiver. With retransmission, if the message still contains errors after decoding using error correction codes, the first packet is combined with the retransmission. The resulting sequence is decoded and checked for errors. If the existence of errors is detected, the second retransmission of the packet is requested. This iterative process is repeated until the latest received packet is error-correctable or the combined copies of the packet result in a sequence which is error-free after error correction. The same updating process described in [5–7] is applicable to this ARQ system as well.

2.3.2.2 Type II ARQ Systems

In this scheme, two codes are used, one of which is an (n, k) block code C_0 for error detection. The other is an invertible half-rate code C_1 which is designed for correction purposes. In this thesis , we use a rate 1/2 memory m Convolutional Code as the invertible code. The information sequence U is fed to the outer Block Encoder, and CRC bits are appended to the sequence. The resulting sequence I is further encoded with Convolutional encoder, and one of the output sequences P_1 or P_2, is then sent over the channel.

If the received sequence, denoted by $\tilde{P}_1$, is decoded successfully, the data is passed to the above layer. However, if $\tilde{P}_1$ contains errors, the receiver sends a NACK back to the transmitter which , upon reception of the NACK (or when a time-out occurs), sends P_2. If $\tilde{P}_2$ is decoded successfully, the data is passed to the above layer. Otherwise, it is combined with $\tilde{P}_1$ and the resulting sequence, $(\tilde{P}_1, \tilde{P}_2)$, is decoded by a rate $\frac{1}{2}$ Viterbi Decoder. If $(\tilde{P}_1, \tilde{P}_2)$ is not decoded successfully, $\tilde{P}_1$ is discarded, and a request for P_1 is made. When $\tilde{P}_1$ is received, it is combined with $\tilde{P}_2$, and If decoding is still unsuccessful, $\tilde{P}_2$ is discarded and a NACK is sent back . This procedure is continued until the complete error recovery is achieved.

Repetition redundancy with error correction codes is used in MARQ schemes to achieve significant improvement in channels with high error rates. For instance, with code combining in the Type-II HARQ, successive copies of the same sequence are stored in the receiver until the error correction power of the combined code is high enough to combat the channel degradation [2].

2.3.2.3 Generalized Self-Decodable (GSD) Type-II Hybrid ARQ Scheme with CPC Codes

In the Type-II Hybrid ARQ scheme with code combining as the channel degrades, the throughput drops sharply to 1/2, and then to 1/3 and so forth [3]. Ideally, one wishes to have an ARQ protocol in which the code rate is adapted to the channel conditions starting with coding rate that overcomes the nominal noise that is always present, and more redundancy is provided as the channel degrades. One such a mechanism is described in [4], and it is referred to as Generalized Self-Decodable (GSD) Type-II Hybrid ARQ Scheme, in which Self-Decodable CPC codes are used.

Let C_i, i = 1, 2, ..., p be p Complementary Punctured Convolutional codes of the same rate. To each k-bit information message to be transmitted, n_p parity bits for error detection and m known tail bits are added. The m tail bits are used to properly terminate the encoder memory and the decoder trellis. The sequence is then encoded with the rate $1/V_0$ code and transmitted according to the following algorithm, in which different stages involved for transmission of a packet are referred to as levels.

level 1: The encoded sequence with code C_1 based on the Perforation matrix P_1 is transmitted. At the receiver, the received sequence is decoded according to the same Perforation matrix. If the CRC indicates an error-free packet, the transmission is complete. Otherwise, the packet is buffered for subsequent decoding attempts and a NACK is sent back to the transmitter.

level 2: Upon the arrival of the NACK, the packet is encoded with code C_2 and Perforation matrix P_2. When this sequence is received, Viterbi Decoding is applied using Perforation matrix P_2. If the decoded information bits are declared error-free, transmission of this packet

is complete. Otherwise, Viterbi Decoding is applied once again, but on the combination of the sequences encoded with codes C_1 and C_2. If the decoding is not successful, the sequence encoded with C_2 is stored in the receiver and a NACK is sent back to the transmitter.

level i: $2 < i \leq p$, The packet is encoded with code C_i, and is transmitted over the channel. The decoding operation is performed on the received packet. If the decoding is not successful, then all the combination of the stored sequences are tried. In the case of decoding failure after code combining operation, the packet encoded with the next code C_i is transmitted. If $i = p$ and the error recovery still can not be obtained, the packet encoded with C_1 is requested and the previously transmitted sequence encoded with C_1 is discarded.

level p+1: When the encoded sequence using C_j, $j = 1, 2, \ldots$ is received, it is decoded and checked for errors. If the decoding is unsuccessful, the decoding resumes using all p sequences available at the receiver. In the event that decoding is still not successful, the next sequence must be repeated. These steps are repeated until the message can be decoded successfully.

2.3.2.4 Generalized Type-II Hybrid ARQ Scheme with Punctured Convolutional Coding

In the Type-II ARQ schemes, whenever a retransmission is required, the transmitter sends a packet of the same length. It follows that the consecutive retransmissions cause the rate of error correction code to be severely penalized. For instance, the code rate is reduced from R to R/2 with only one retransmission. This is considered as a drastic drop in the rate which may not be justifiable. A scheme which has the ability to adapt its rate much more smoothly is referred to the Generalized Type-II Hybrid ARQ System. It is based on the Punctured Convolutional Codes and is discussed in Kallel and Hagenauer's papers [14], [15]. This

scheme makes use of the Rate Compatible Punctured and Repetition codes for successive parity retransmission to build up a code which is powerful enough to decode the message. When transmitting a sequence of information, the following steps are performed. To each k-bit message to be transmitted, n_p parity bits and m known bits are added. The resulting sequence is encoded with the rate $1/V_0$ Convolutional Code and stored at the transmitter. The transmission strategy, then, is performed as follows:

1. The encoded sequence according to a starting perforation matrix P_i is transmitted.
2. The received sequence is decoded using the Viterbi algorithm based on the same perforation matrix.
3. Error detection is performed on the decoded message. With the detection of errors, the steps (1)-(3) are repeated but with each retransmission of the same packet a new perforation matrix P_i, P_{i+1}, ... , corresponding to a lower-rate code of a family of rate-compatible codes is employed to determine the extra bits required for that transmission. In other words, the redundancy in each repeated packet depends on the difference between the previous and current perforation matrices.

If the decoding is still not successful, Rate-Compatible Repetition Codes [14] are employed until the decoding finally succeeds.

2.4 Frame Structure

In data communications all the transmissions are made in frame format. The purpose of this framing is to decide where successive frames start and stop. In the High level Data Link Control (HDLC), the flag and other parts of the frame are distinguished using zero

insertion and deletion operation referred to as bit-stuffing. When the BER becomes high, synchronization problems arise and the flag pattern is sometimes generated within the data weakening the zero insertion and deletion's effect. So, it is almost impossible to use the HDLC flag as it is, for a data communication protocol with poor channel conditions because it leads to rapid deterioration of the ARQ system performance.

As a substitute method, a frame synchronization technique described in [25] is adopted. This technique requires the transmission of frames with the structure shown in figure 2.4. Each frame begins with a frame synchronization pattern. Following that is the header which

Figure 2.4 Format of the Transmitted Frame

contains the routing information, the sequence and acknowledgment numbers, and the length of the packet. The information bits form the last part of the frame.

2.4.1 Flag Field: A unique frame synchronization pattern of length F-bits is included at the beginning of each frame to identify the start of the frame. The end of the frame is determined from the length field included in the header. An efficient technique for detection of this pattern at the receiver is investigated in the next section.

2.4.2 Header Field: The control bits for network applications and routing information constitute the header, which is encoded separately form the data. In this thesis, Punctured and Repetition Convolutional Codes are utilized for encoding the header. The address field is used to identify the transmitter and the receiver. This field is not needed for point-to-point link but it is always included for uniformity. The control bits are the sequence numbers and

other necessary information crucial for processing the information packet. The last part of the header indicates where the frame ends.

The reasons for separate encoding/decoding operations for the header are four folds. One, the routing information must be known before deciding to accept a packet. Two, the sequence number of the received packet enables the receiver to determine out-of-sequence packets, saving the amount of time spent on decoding the rest of the data frame. Three, in adaptive ARQ strategies, the header indicates which of the sequences and what code rate are used so that correct measures can be taken at the receiver. Four, frame length is included for reliable end of frame identification.

As explained in the preceding paragraph, the frame header comprises the essential part of the frame because it includes all the necessary information required for processing the data packets. It must be protected in such a way that its effect on the system performance can be kept at a minimum level. Means of achieving this goal will be proposed and investigated in the next chapter.

2.4.3 Information Field: This field can contain any sequence of bits, but generally limited to a specified maximum depending on the application. It is encoded with an error detection code followed by a Convolutional Codes for error correction purpose depending on the ARQ system being employed.

2.5 Frame Synchronization

The ARQ techniques depend on reliable frame synchronization mechanism. In this section and the rest of the thesis, we assume an efficient algorithm for frame synchronization

which is proposed by Driessen [25]. This scheme makes careful selection of the synchronization pattern to minimize the correlation side-lobes and the probability of false synchronization. In this scheme, the frame synchronization pattern of F-bit is embedded in the data stream and is preceded by a preamble of the same length for bit synchronization. Figure 2.5 shows the structure of the transmitted sequence of length $F = MT + N + L$, consisting of three subsequences: the periodic preamble word P of length MT and period T, the synchronization pattern a^N of length N and the rest of the data frame d^L of length L.

Figure 2.5 Structure of the Transmitted Sequence

This scheme performs a sliding correlation of the received data stream with the known pattern a^N. The beginning of the data frame is declared if the peak value of correlation exceeds a certain threshold. The selected pattern a^N must be chosen such that the output of the sliding correlator C_k is low while the preamble and the first $N-1$ digits of the synchronization pattern are received.

$$C_k = \sum_{i=0}^{N-1} a_i p_{i+k}, \qquad k \leq 0 \tag{2.12}$$

In other words, low correlation side-lobes are desired when the pattern is preceded by a preamble. The subsequent C_k where $0 \leq k \leq N-1$ represents the correlation between the concatenated preamble/frame synchronization pattern.

$$C_k = \sum_{i=0}^{N-1-k} a_i p_{i+k} + \sum_{i=N-k}^{N-1} a_i a_{i+k-N}, \qquad 0 \leq k \leq N-1 \tag{2.13}$$

If the value of the correlation is greater than or equal to a threshold, then it is declared that the frame synchronization is accomplished. The probability of success and false synchronization [26] are given by

$$\begin{aligned} P_s &= \sum_{k=0}^{(N-Th)/2} \left[(1/2)^N \sum_{n=k}^{N} \binom{N}{n} \right]^{F-1} \binom{N}{k} p_b^k (1-p_b)^{N-k}, \\ P_f &\leq (F-1)(1/2)^N \sum_{k=0}^{N} \binom{N}{k} p_b^k (1-p_b)^{N-k} \left[\sum_{n=0}^{min(k,(N-Th)/2)} \binom{N}{n} \right], \end{aligned} \tag{2.14}$$

where p_b is the channel bit error rate. Optimization of P_s and P_f can be obtained simultaneously . For instance, to lower the probability of false synchronization, it is desirable to have a large value for threshold. However, large values of threshold leads to a decrease in the probability of success. Numerical results in figures 2.6 and 2.7 show the success and false alarm probabilities for the synchronization pattern 2941b3 in hexadecimal when preceded by preamble "1010..." of length 24 and period 2. The length of the data frame is configured at 1024 bits in this investigation. The threshold value of 12 seems to be the optimum value since the false alarm probabilities are still low. Simulated results in figure 2.8 indicate the validity of the above argument.

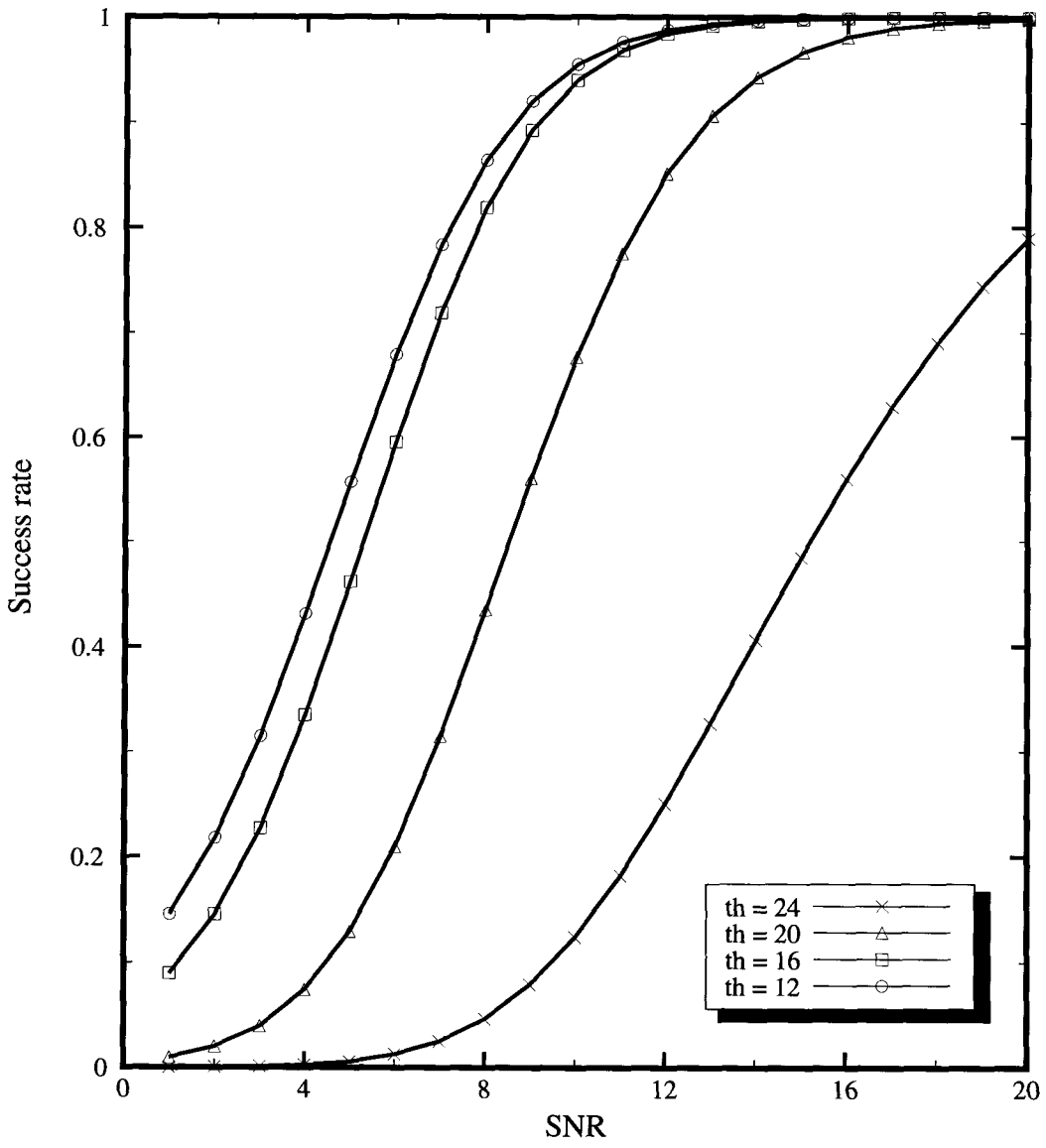

Figure 2.6 Probability of Successful Synchronization

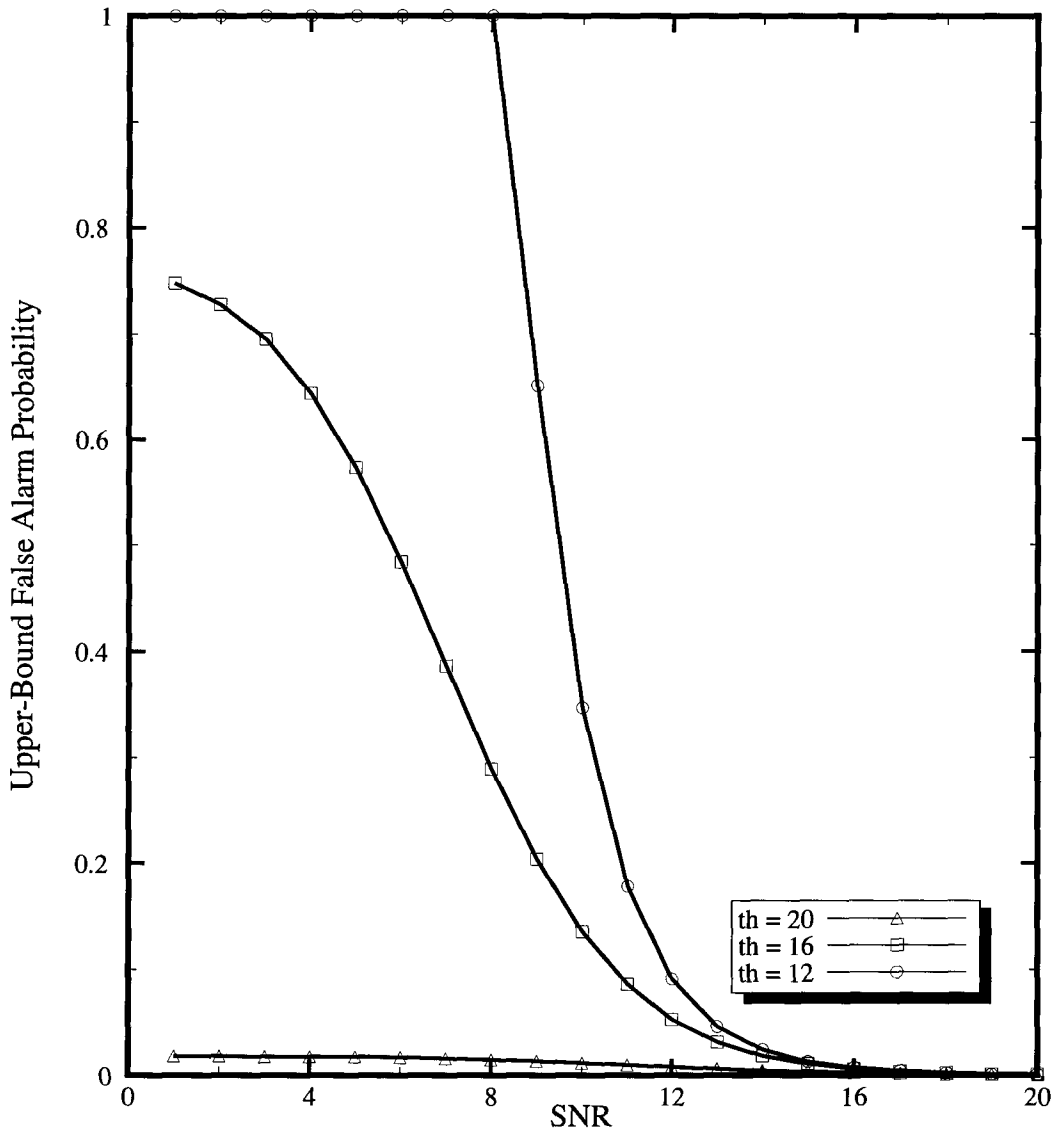

Figure 2.7 False Alarm Probability for Synchronization Word 2941b3

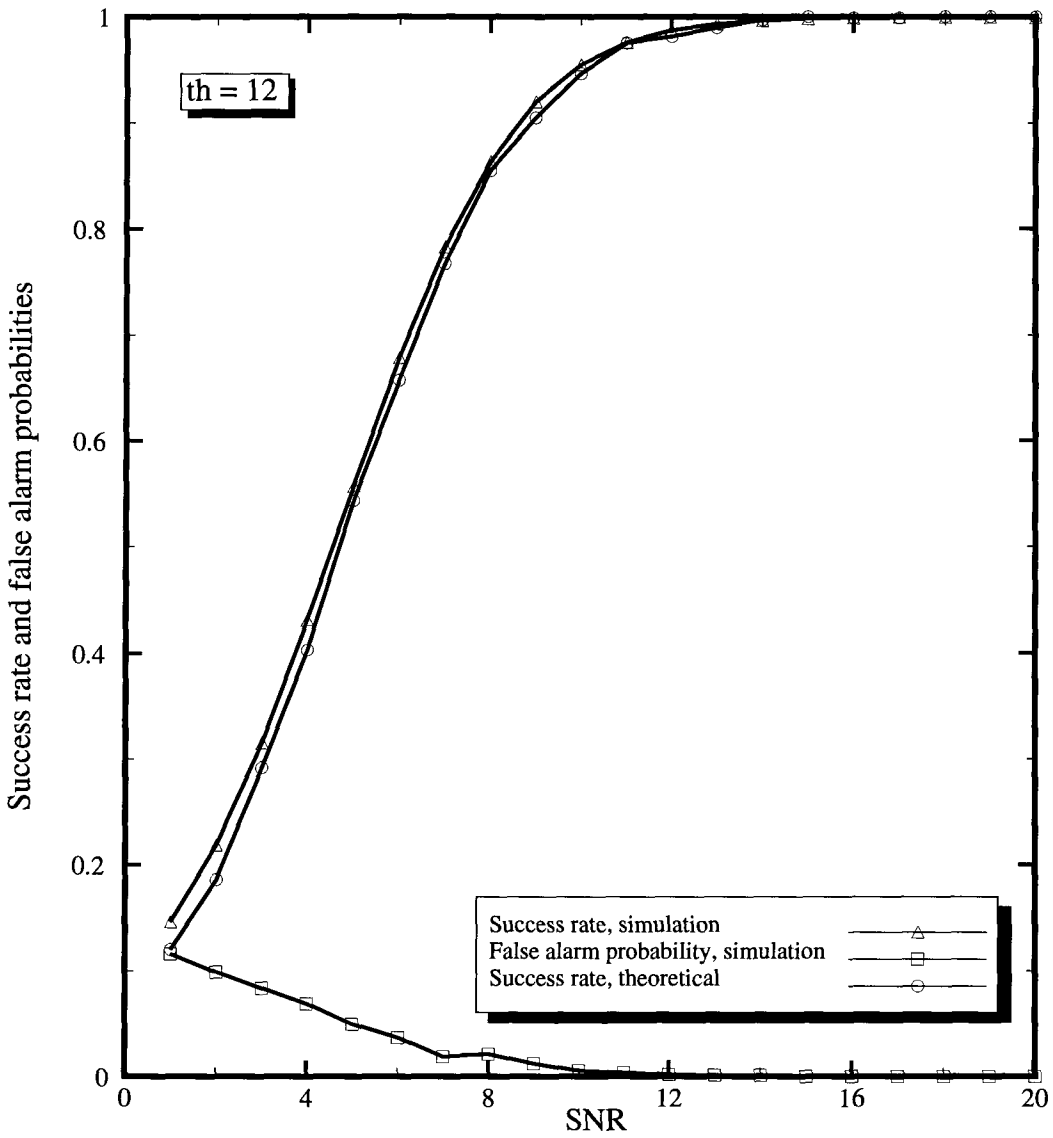

Figure 2.8 Simulated Success and False Alarm probability

Chapter 3 Effects of Imperfect Feedback Channel and Frame Header Sensitivity on the Performance of Memory ARQ Schemes

When the frame header fails, the transmission is essentially wasted. The feedback channel noise can also result in the occurrence of unnecessary transmissions. ARQ protocols are typically analyzed without the consideration of these effects, and it is usually assumed the return channel is noiseless. However, there are circumstances such as time-varying channels and Memory ARQ (MARQ), in which these simplifying assumptions are not valid, and a more realistic evaluation of the system performance is required. The influence of these parameters can be so drastic that MARQ can become completely obsolete.

At high channel error rates, the amount of repetition redundancy available at the receiver, in MARQ systems, can be substantially reduced due to frequent frame header loss. The consequence is low probability of successful error correction using either reliability updating process [7], or majority voting rule [6], or repetition with error correction codes [2] depending on the MARQ being studied. Therefore, the advantage of MARQ, allowing exploitation of retransmissions of a packet for facilitating correct error recovery, can not be completely achievable.

A general investigation of the performance of the Memory ARQ (MARQ) schemes with noisy feedback channel and frame header sensitivity is carried out, and the corresponding throughput expressions are developed. Numerical examples are given to indicate how the throughput can be affected when these parameters are considered in the analytical procedure. Simulations are also run to illustrate the validity of the proposed algorithm. Some techniques

are proposed to counteract the effects of these parameters without affecting the overall system performance.

3.1 General Description of Memory ARQ Schemes (MARQ)

MARQ utilize the redundancy provided by retransmissions of a packet to reduce the average number of transmissions. In this section, a general algorithm describing these MARQ schemes is studied, and it is demonstrated that it is feasible to derive an expression for the throughput performance while the effects of the ACK and frame header sensitivities are taken into consideration. It is shown that all the ARQ schemes are the special cases of this General Scheme, since they are obtained by setting some of the system parameters to specific values.

The message is contained within a frame and preceded by the frame header. The frame header is encoded separately to extract the required information before deciding to process the following data section. The ACK signal regarding the completion status of a data packet is transmitted via a return channel, and it is subject to being damaged due to the return channel errors.

For the General Scheme, there are t possible codes C_i, i =1...t, with rates R_i, i=1...t. The employed codes are self-decodable, and their corresponding rates R_i are all equal. Moreover, it is possible to combine different groups of these codes to form new codes which possess higher error-correcting capability. The algorithm for this scheme, in which levels are used to represent the different transmission stages, is as follows:

Level 1: The encoded sequence with rate R_1 is transmitted. If the CRC indicates an error-free message, the transmission is complete. Otherwise, the packet is buffered for subsequent

decoding attempts and a NACK is sent back to the transmitter.

Level 2: Upon the arrival of the NACK, the packet is encoded with rate R_2 and sent to the receiver. At the receiver, the incoming sequence is decoded and examined for presence of errors. If the decoded sequence contains no error, it is delivered to the user. Otherwise, the decoding operation is performed on the combined sequences transmitted with rates R_1 and R_2 respectively. If the resulting code still does not have the error-correcting capability required to combat the channel noise, the system moves up to level 3.

level i: $2 < i \leq t$. The encoded packet with rate R_i is transmitted over the channel. The decoding operation is performed on the received packet. With the detection of errors in the packet, the decoding process is repeated using all the i sequences available at the receiver. In the case of unsuccessful decoding after this code combining operation, the packet encoded with the next rate R_i is repeated. If $i = t$ and the error recovery still can not be obtained, the packet encoded with rate R_1 is requested and the previously stored sequence transmitted with rate R_1 is discarded.

Level t+1: When the encoded packet with rate R_j, $j = 1, 2, \ldots$ is received, it is decoded and checked for errors. If the decoding is unsuccessful, the decoding resumes using all t sequences available at the receiver. In the event, that the decoding is still not successful, the encoded packet with the next rate is requested.

Thus, the sequences are updated one at a time to utilize all the stored sequences for proper code combining operation. After 2t transmissions, the whole set of sequences will have been repeated.

3.1.1 Throughput Analysis

In this section, we will analyze the throughput performance for the General Scheme in Stop-and Wait or Selective-Repeat mode with an infinite receiver buffer. The throughput analysis is based on the assumption that the forward and return channel noise are randomly distributed.

As it is clearly inferred from the description of the algorithm, there are dependencies among the joint events in the different states which make the performance analysis of the original scheme difficult. Thus, to facilitate the analysis of the throughput efficiency, we adopt an inferior system approach, the throughput of which will serve as a lower bound for the performance of the original system.

This inferior system operates in the same manner as the original scheme except after t transmissions, all the erroneous copies of a packet are discarded, and the transmission is repeated form level 1. Thus, the retransmissions will be packets encoded with codes having rates (R_1, R_2, ..., R_t), (R_1, R_2, ..., R_t), ..., assuming successive NACK's or time-outs. The time-out for a packet occurs if the frame header or the signal about the completion status of a packet in the return channel is lost.

It is obvious that this system is inferior to the original system as far as the throughput performance is concerned. Unlike the original system which updates the codes one by one after level t, this inferior system discards the stored sequences and renews the whole set when the retransmissions are still requested after level t. Therefore, the evaluation of probability of decoding event depends only on the limited number of codes.

The state diagram of the inferior scheme is shown in Figure 3.1, in which we let t

= 3 to simplify the description of the diagram. Before describing the algorithm using the state diagram, we need to define some of the parameters which are used in the throughput derivation of the inferior system.

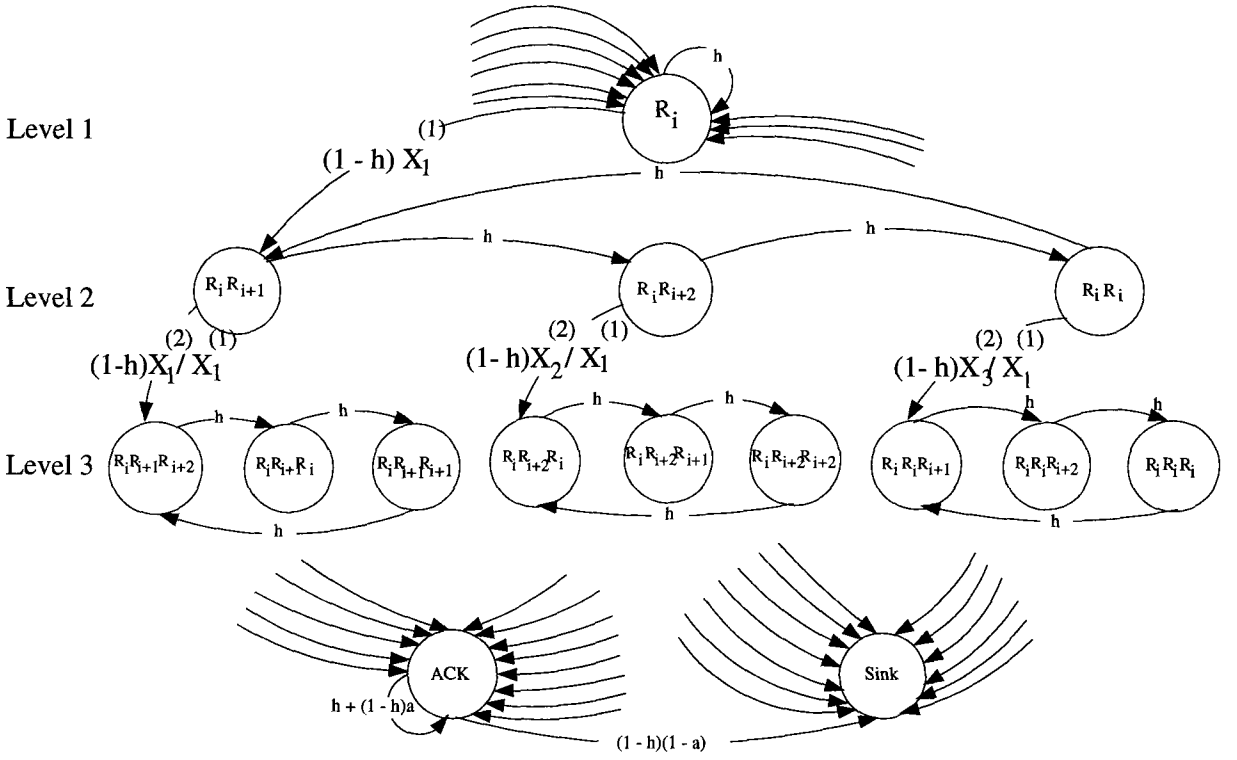

Figure 3.1 State Diagram of the Inferior System with ACK and Frame Header Sensitivity

The missed ACK and frame header can result in time-out for the buffered packet in the transmitter and consequently redundant retransmissions. The ACK and frame header failure probabilities are represented by **a** and **h** respectively. It should be noted that the effect of a lost NACK is eliminated from the analysis since in the case of a NACK another retransmission is required anyway due to the packet being in error. Therefore, whether the NACK is missed or received successfully does not affect the average number of transmissions required.

Let $D^{(R_i)}$ be the event that decoding with rate R_i fails to recover the message; $D^{(R_i,R_{i+1})}$ is the event that decoding of the combined codes C_i and C_{i+1} with the corresponding rates R_i and R_{i+1} produces an erroneous message. $D^{(R_i,R_{i+2})}$ is the event that the resulting code from the combined codes C_i and C_{i+2} is not powerful enough to be decoded successfully. $D^{(R_i,R_i)}$ is the event that combining the two corrupted copies of the code C_i can not lead to successful recovery of the message. Similarly, the decoding failure events for resulting codes from different combination of three codes are defined in the same manner. Then, we can write and simplify all the joint event probabilities as follows:

$$\begin{aligned} P\Big(D^{(R_i)}, D^{(R_{i+1})}, D^{(R_i,R_{i+1})}\Big) &\leq P\Big(D^{(R_i,R_{i+1})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_{i+2})}, D^{(R_i,R_{i+2})}\Big) &\leq P\Big(D^{(R_i,R_{i+2})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_i)}, D^{(R_i,R_i)}\Big) &\leq P\Big(D^{(R_i,R_i)}\Big), \end{aligned} \tag{3.1}$$

$$\begin{aligned} P\Big(D^{(R_i)}, D^{(R_{i+1})}, D^{(R_i,R_{i+1})}, D^{(R_{i+2})}, D^{(R_i,R_{i+1},R_{i+2})}\Big) &\leq P\Big(D^{(R_i,R_{i+1},R_{i+2})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_{i+1})}, D^{(R_i,R_{i+1})}, D^{(R_i)}, D^{(R_i,R_{i+1},R_i)}\Big) &\leq P\Big(D^{(R_i,R_{i+1},R_i)}\Big), \\ P\Big(D^{(R_i)}, D^{(R_{i+1})}, D^{(R_i,R_{i+1})}, D^{(R_{i+1})}, D^{(R_i,R_{i+1},R_{i+1})}\Big) &\leq P\Big(D^{(R_i,R_{i+1},R_{i+1})}\Big), \end{aligned} \tag{3.2}$$

$$\begin{aligned} P\Big(D^{(R_i)}, D^{(R_{i+2})}, D^{(R_i,R_{i+2})}, D^{(R_i)}, D^{(R_i,R_{i+2},R_i)}\Big) &\leq P\Big(D^{(R_i,R_{i+2},R_i)}\Big), \\ P\Big(D^{(R_i)}, D^{(R_{i+2})}, D^{(R_i,R_{i+2})}, D^{(R_{i+1})}, D^{(R_i,R_{i+2},R_{i+1})}\Big) &\leq P\Big(D^{(R_i,R_{i+2},R_{i+1})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_{i+2})}, D^{(R_i,R_{i+2})}, D^{(R_{i+2})}, D^{(R_i,R_{i+2},R_{i+2})}\Big) &\leq P\Big(D^{(R_i,R_{i+2},R_{i+2})}\Big), \end{aligned} \tag{3.3}$$

$$\begin{aligned} P\Big(D^{(R_i)}, D^{(R_i)}, D^{(R_i,R_i)}, D^{(R_{i+1})}, D^{(R_i,R_i,R_{i+1})}\Big) &\leq P\Big(D^{(R_i,R_i,R_{i+1})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_i)}, D^{(R_i,R_i)}, D^{(R_{i+2})}, D^{(R_i,R_i,R_{i+2})}\Big) &\leq P\Big(D^{(R_i,R_i,R_{i+2})}\Big), \\ P\Big(D^{(R_i)}, D^{(R_i)}, D^{(R_i,R_i)}, D^{(R_i)}, D^{(R_i,R_i,R_i)}\Big) &\leq P\Big(D^{(R_i,R_i,R_i)}\Big). \end{aligned} \tag{3.4}$$

For simplicity of notations, we let

$$
\begin{aligned}
&X_1^{(1)} = P\left(D^{(\mathrm{R_i})}\right), \\
&X_1^{(2)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+1}})}\right),\ X_2^{(2)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+2}})}\right), X_{(3)}^{(2)} = P\left(D^{(\mathrm{R_i},\mathrm{R_i})}\right), \\
&X_1^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+1}},\mathrm{R_{i+2}})}\right),\ X_2^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+1}},\mathrm{R_i})}\right),\ X_3^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+1}},\mathrm{R_{i+1}})}\right), \\
&X_4^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+2}},\mathrm{R_i})}\right),\ X_5^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+2}},\mathrm{R_{i+1}})}\right),\ X_6^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_{i+2}},\mathrm{R_{i+2}})}\right), \\
&X_7^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_i},\mathrm{R_{i+1}})}\right),\ X_8^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_i},\mathrm{R_{i+2}})}\right),\ X_9^{(3)} = P\left(D^{(\mathrm{R_i},\mathrm{R_i},\mathrm{R_i})}\right),
\end{aligned}
\tag{3.5}
$$

where the superscripts and subscripts indicate the level and the corresponding state in a level respectively. These variables associate to the transition probabilities from level 1 to level 2 and 3. The transmission algorithm always starts form the state at level 1, and the packet transmission ends in one of the states of the diagram with successful reception of the message. The first transmission is the sequence based on the code with R_1. On arrival of a frame, if the frame header fails, the time-out occurs, and the retransmission consists of a code sequence with rate R_2. Again, if the preceding problem prevails, the system remains in the same state and repeats the message with different code rates R_i until the frame header is successfully received. The received sequence is, then, decoded and checked for errors. If the decoded message contains error, the erroneous packet is saved and a retransmission is requested. The probability of decoding error for the received sequence after the successful reception of the frame header is denoted by (1–h)$X_1^{(1)}$ which is indicated as the transition probability from the state at level 1 to the state at level 2.

The retransmission of the message is carried out at level 2, and it is the message encoded with the next rate. For instance, if the sequence based on R_i was sent before, the sequence

with the rate R_{i+1} will be the next transmission. Being in any of the states in level 2, we change to level 3 if the frame header is successfully decoded and the error-correcting capability of the two combined codes is not sufficient to recover the message. Therefore, the packet transmission will be at level 3 if decoding fails for the first sequence and the decoded message from the combined codes at level 2 contains errors. The next retransmission after level 3 takes place in level 1 according to the description of the inferior system.

As the transmission proceeds, the depth changes with the presence of errors in the decoded sequences. It was verified that the probability of decoding error at this level had to be divided by that of the unsuccessful decoding probability in the previous level to ensure that the right terms were obtained with each depth change. For instance, the system reaches level 3 from level 1 if the decoding attempt for the encoded packet with rate R_i fails and the resulting code from the combined codes R_i and R_{i+1} does not yield an error-free message. This transition probability is given by

$$\begin{aligned} P\left(D^{(R_i)}, D^{(R_{i+1})}, D^{(R_i,R_{i+1})}\right) &\leq P\left(D^{(R_i,R_{i+1})}\right) \\ &= X_1^{(2)}. \end{aligned} \tag{3.6}$$

To obtain $X_1^{(2)}$ by making transition from level 1 to level 3 in the state diagram, we must divide it by the previous decoding error probability denoted by $X_1^{(1)}$. All other terms, which are shown in the diagram as the transition probabilities from one level to the next, are obtained using the same procedure. The transition probabilities from level 3 to level 1 and their corresponding branches are not indicated in the diagram to keep it simple to understand.

Two extra branches (not shown in the diagram) emanate form each state, one of which terminates at the ACK state and the other ends at the Sink state. From the current state, the

transition is made to the Sink state if the transmitted message is successfully acknowledged, whereas the ACK state is reached when the ACK is detected in error or lost. In the ACK state, unnecessary transmissions occur because the transmitter is not aware of the successfully received packet at the receiver. When a frame is retransmitted due to a missed ACK, the receiver attempts to detect the header. If the decoding of header is unsuccessful, the time-out for the frame occurs and consequently the frame is retransmitted again. If the frame header is recovered successfully and the ACK message is lost, the scheme stays in the ACK state. When the ACK message is successfully received, the transmission is complete and the system moves to the Sink state.

The number of states depends on the number of codes used in the algorithm. For instance, when c codes are available for encoding a message, the total number of states is

$$n = \sum_{j=1}^{t} c^{(j-1)} + 2, \tag{3.7}$$

where t is the number of levels and the constant 2 is due to the Sink and the ACK states. For the cases when all the c codes are exactly the same, the total number of states becomes

$$n = t + 2. \tag{3.8}$$

This process is a Markov chain because being in state i, there is a fixed transition probability P(j | i) that the system will be in state j. The transition probabilities all add up to 1 when the sink state is included in the state diagram. We can arrange these transition probabilities into an $n \times n$ matrix, G, where the element in the i-th row and j-th column

is P(j | i).

$$G = \begin{bmatrix} \mathrm{h} & (1-\mathrm{h})X_1^{(1)} & \dots & \dots & (1-\mathrm{h})\left(1-X_1^{(1)}\right)\mathrm{a} & (1-\mathrm{h})\left(1-X_1^{(1)}\right)(1-\mathrm{a}) \\ 0 & \mathrm{h} & 0 & \dots & (1-\mathrm{h})\left(1-X_1^{(2)}/X_1^{(1)}\right)\mathrm{a} & (1-\mathrm{h})\left(1-X_1^{(2)}/X_1^{(1)}\right)(1-\mathrm{a}) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \mathrm{h}+(1-\mathrm{h})\mathrm{a} & (1-\mathrm{h})(1-\mathrm{a}) \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

The transition probabilities from the Sink state to other states are given by

$$\begin{aligned} P(j \mid \text{Sink}) &= 0, \quad for \;\; 1 \leq j < n, \\ P(\text{Sink} \mid \text{Sink}) &= 1, \end{aligned} \tag{3.10}$$

which indicate that there is no transition from this state back to any other states in the diagram. In other words, this state is totally unused.

To obtain an expression for the expected number of transmissions, we must only take into account the transmissions due to unsuccessful reception or decoding failure of a packet but transitions to the Sink state are indications of the successful recovery of message. Therefore, we must exclude the effect of these transition probabilities by eliminating the n-th row and n-th column from G matrix, resulting in another sub-matrix denoted by G', which has $(n-1) \times (n-1)$ elements.

Let's define the vector, W, to be an (n–1)-component column vector whose i-th component is w_i. Each element of this vector is used to represent the conditional probability that a certain event occurs while in state i (such events are indicated in the states). The expected number of occurrence of such a certain event indicates how many times this event is repeated and the state i is revisited. Also, let V be the initial state probability row vector with (n–1) components indicating the chance of being in a state at the start of transmission.

The expected number of transmissions for a given frame in the General Scheme is, then, upper bounded by

$$\bar{N} \leq 1 + \mathrm{V}\sum_{i=1}^{\infty} [G']^i \mathrm{W} = \mathrm{V}(I - G')^{-1}\mathbf{W}, \tag{3.11}$$

where G' [27] is the sub-matrix of G which is a stochastic matrix, and the vector W is a unit vector. As for the initial state probabilities represented by V, they are all 0 except the first component since the transmission always starts from the state at level 1. i.e V = [1 0 0 0 ...].

The throughput efficiency is defined as the ratio of the average number of information bits per transmitted channel symbol. Therefore, the lower bound for the normalized throughput efficiency of the original system can be obtained as follows:

$$\eta \geq \frac{k}{(k + header\ bits + redundant\ bits)\bar{N}} \tag{3.12}$$

where the factor *k /(k + header bits + redundant bits)* represents the loss in throughput due to the overhead included in the transmitted sequence.

3.2 Application of the Throughput Derivation to the Most Common MARQ Protocols

3.2.1 Type–0 ARQ Scheme with Copy Combining

Setting t = 2 and $R_1 = R_2 = 1$ in the general algorithm for MARQ Schemes, we obtain the Type-0 ARQ scheme with two copy-combining. The inferior system for this scheme performs error detection when the first copy of a packet is received. With the arrival of the second copy of the same packet, copy combining is performed. If the data recovery still can not be obtained, both copies of the packet are discarded.

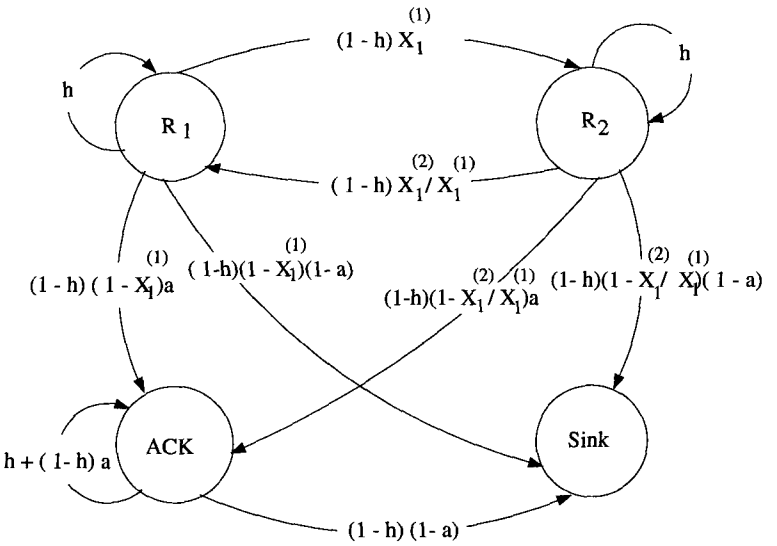

Figure 3.2 Markov chain for the Type-0/I MARQ Scheme with Two-copy Combining

The Markov chain in Figure 3.2 is constructed based on the proposed inferior system, where two-copy combining is assumed. The transmission initiates with the system being in the state R_1 in the far left-hand corner. If the header of the received sequence is correct and the data field contains errors, a transition to the second state R_2 is made. With the arrival of the second copy of the same packet, copy-combining operation is performed. If the message still can not be recovered, the system discards the stored sequences and moves to the previous state R_1 and the transmission starts all over again.

Then, the transition matrix G' for this scheme is given by

$$G' = \begin{bmatrix} \mathrm{h} & (1-\mathrm{h})X_1^{(1)} & (1-\mathrm{h})\left(1-X_1^{(1)}\right)\mathrm{a} \\ (1-\mathrm{h})X_1^{(2)}/X_1^{(1)} & \mathrm{h} & (1-\mathrm{h})\left(1-X_1^{(2)}/X_1^{(1)}\right)\mathrm{a} \\ 0 & 0 & \mathrm{h}+(1-h)\mathrm{a} \end{bmatrix} \tag{3.13}$$

and the expected number of transmissions is upper bounded by

$$\begin{aligned}\bar{N} &\leq 1 + \mathrm{V}\sum_{i=1}^{\infty}\left[G'\right]^{i}\mathrm{W} = \mathrm{V}\left(I - G'\right)^{-1}\mathrm{W} \\ &= \frac{1 + X_1^{(1)}}{(1-\mathrm{h})\left(1 - X_1^{(2)}\right)} + \frac{\mathrm{a}}{(1-\mathrm{a})(1-\mathrm{h})}\end{aligned} \tag{3.14}$$

Let L be the length of the transmitted sequence, then $X_1^{(1)}$ and $X_1^{(2)}$ are given by

$$\begin{aligned}X_1^{(1)} &= 1 - (1-p)^L, \\ X_1^{(2)} &= 1 - (1-p_2)^L,\end{aligned} \tag{3.15}$$

where p is the channel error rate and p_2 is obtained from Equation (2.2) with d= 2. The packet length is 900 bits throughout this chapter with an optimum block code of 32 bits for error detection to make the probability of undetected error small.

Figure 3.3 shows the comparative system performance of the simple ARQ and Type-0 MARQ scheme with two copy combining when the influence of the frame header is taken into account. In Figure 3.3 the ideal-case throughput curve corresponds to the situation where all the transmitted frames can reach the intended recipient without being lost. In other words, the failure probabilities for the frame header and ACK message are equal to zero. In such a case, maximum utilization of the available repetition redundancy at the receiver can be made. The numerical results for the ideal-case scenario are obtained by setting **h** and **a** to zero in the transition matrix G'. The numerical results for the scheme with frame header sensitivity and noise-free feedback channel (a = 0) are also calculated. **h** is obtained from

$$\mathrm{h} = 1 - (1-p)^{L_{\mathrm{h}}}, \tag{3.16}$$

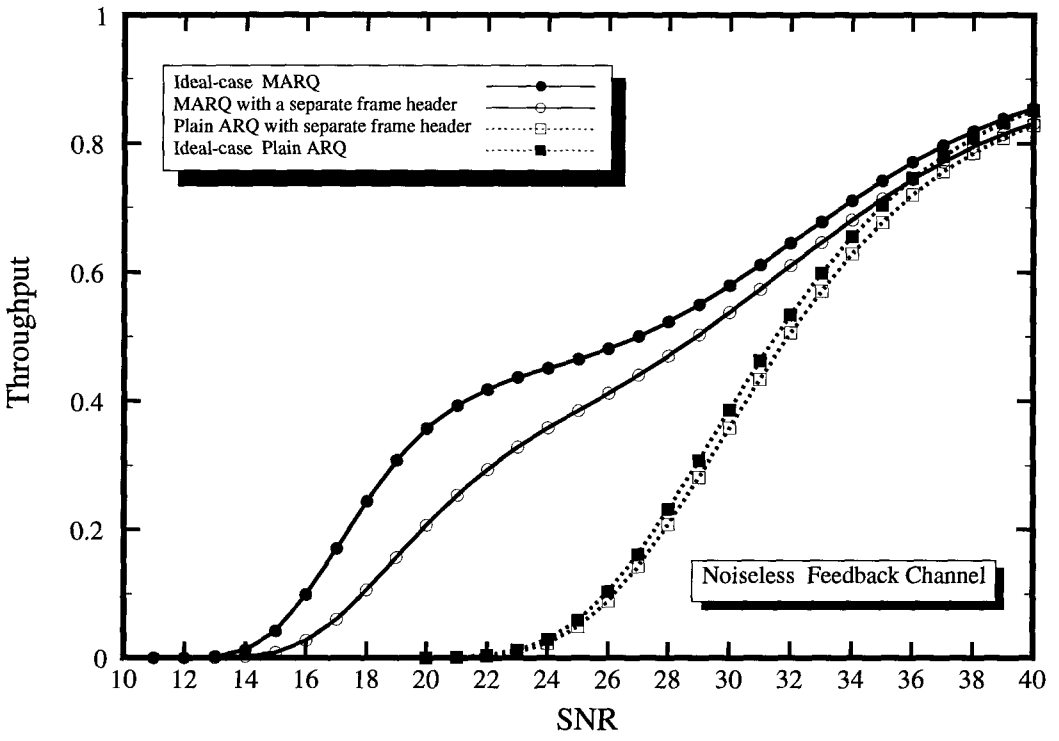

Figure 3.3 Effect of Frame Header on the Throughput of Simple ARQ and a MARQ Systems

where L_h is the length of the frame header which is set to 57 bits. The numerical results for the simple Type-0 ARQ are calculated from

$$\bar{N} = \frac{1}{(1-\mathrm{h})\left(1-X_1^{(1)}\right)} + \frac{\mathrm{a}}{(1-\mathrm{h})(1-\mathrm{a})} \tag{3.17}$$

which is obtained by letting t = 1 and R_t=1 in the General Scheme.

It is clearly evident that the performance of MARQ schemes are severely affected when the frame header failure rate is high. These numerical results prove the validity of the statements made earlier in the chapter regarding the significance of the header. In other words, missed headers can counteract the advantage of MARQ which use repetition redundancy for effective error control. Although the exploitation of corrupted copies of a packet is allowed

in these schemes to enhance efficiency, this process can not be performed when the frame header is lost which also implies the packet loss.

To ensure achievement of acceptable throughput values at low signal-to-noise ratio, we include two consecutive headers at the beginning of each transmitted frame and combine them when they are received at the receiver. The header failure rate is, then, calculated from

$$\mathrm{h} = 1 - (1 - p_2)^{L_\mathrm{h}}. \tag{3.18}$$

Figure 3.4 shows the system performance of the scheme with copy combining for both the header and the data field in a noiseless return channel. It is observed that the influence of unprotected frame header brings about unsatisfactory throughput at high channel error rates. For instance, throughput degradation of up to 50% is caused at 22 dB. However, when two consecutive headers are included in the data frame and combined at the receiver, the efficiency of the system suffers slightly at very high SNR values due to extra redundancy transmitted but the performance degradation is lowered by 35% at 22 dB. In this way, the errors introduced by the channel are often corrected automatically by copy combining of the sequences.

We discovered that the throughput improvement could be achieved if we allowed the utilization of copy combining for the header in the same way as performed for the data section. Therefore, we apply this technique to the frame in the forward channel and investigate the effect that noisy return channel may have on the protocol performance. The throughput performance of the scheme in such a scenario is shown in Figure 3.5. It is assumed that the ACK signal is contained in a frame header which has the same structure as that in the

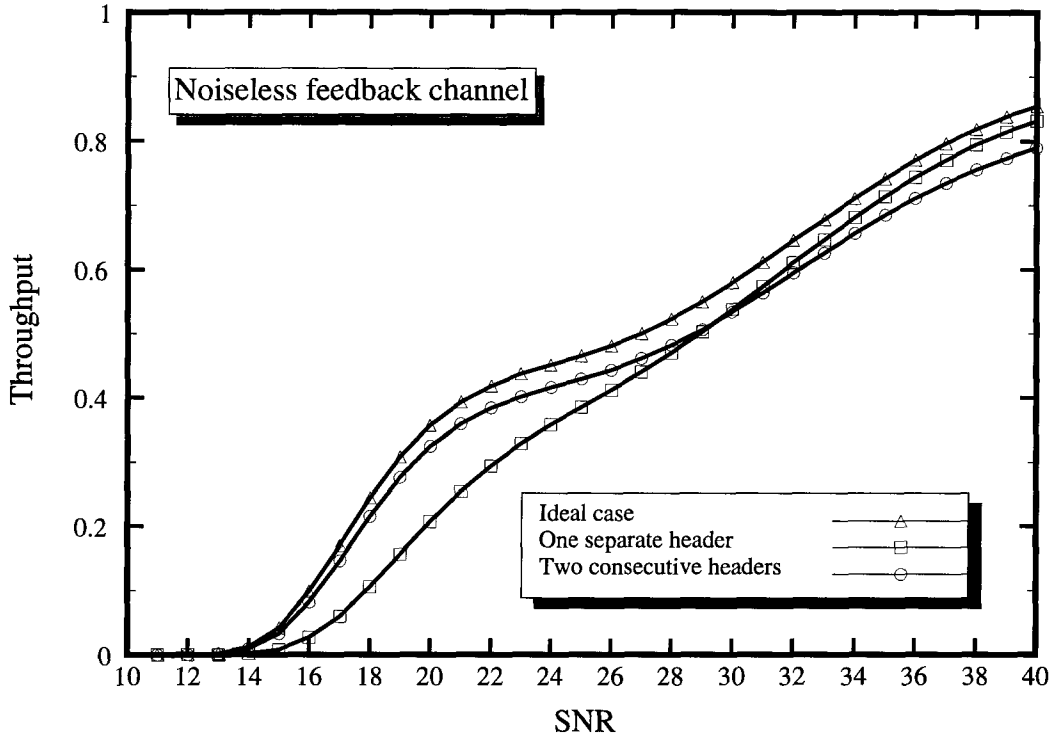

Figure 3.4 Throughput of the Type-0 ARQ Scheme with Copy Combining

forward channel. The ACK and header failure probabilities are calculated from

$$\begin{aligned} \mathrm{h} &= 1-(1-p_2)^{L_\mathrm{h}}, \\ \mathrm{a} &= 1-(1-p)^{L_\mathrm{a}}, \end{aligned} \tag{3.19}$$

where header and ACK lengths are denoted by L_h and L_a. p is the channel error rate and p_2 is obtained from Equation (2.2) with d = 2. The ACK is assumed to have the same length as the header. It is seen that if only one header is used in the return channel, the difference in performance for noisy and noiseless feedback channels is about 30% at 22 dB. Similar to the preceding investigation for the frame header in the forward channel, if the packet in the return channel is preceded with two consecutive frame headers, the algorithm of the MARQ can operate close to optimal with respect to the lower bound on the throughput. In such a

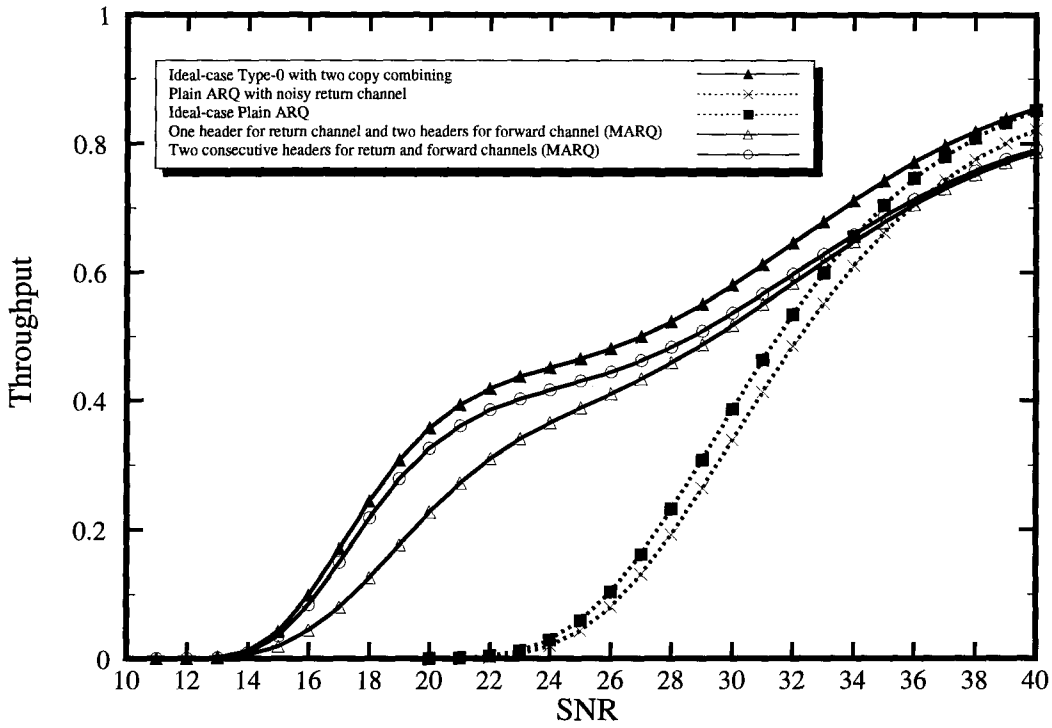

Figure 3.5 Throughput of the Type-0 ARQ Scheme with Copy Combining

case, the performance curves overlap and this optimal performance is maintained over the most useful range of SNR values.

3.2.2 Type-II ARQ Schemes

The derivation of the performance expressions for the conventional Type-II ARQ scheme is similar to that of the Type-0 with copy combining with the difference that this scheme uses error-correction codes. This protocol is a special case of the general MARQ scheme with $t = 2$, $R_1 = 1$, and $R_2 = 1$.

The inferior system for this scheme works as follows: Assume that two sequences with rates R_1 and R_2 are used for transmission. When the sequence with rate R_1 is received, it is used to recover the corresponding data. If it is detected in error, it is stored in the

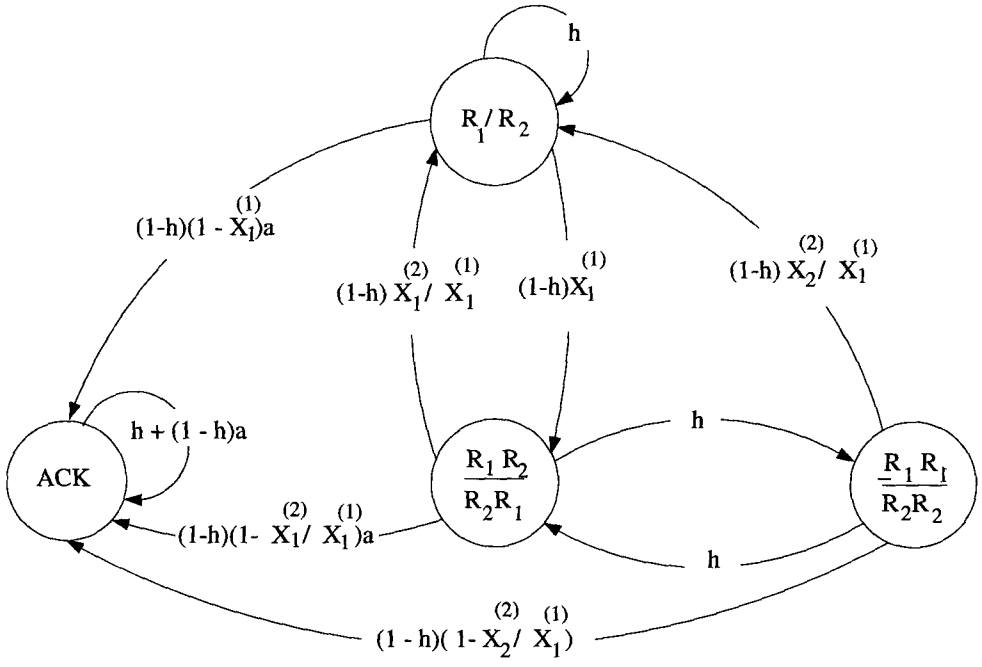

Figure 3.6 Markov Chain Model for the Modified Type-II HARQ Scheme

receiver. With the arrival of the encoded sequence with rate R_2, if errors are detected, Viterbi Decoding is performed on the combined sequences (R_1, R_2). If the decoding fails to recover the message, then both sequences are discarded. Also, if two copies of the sequence encoded with the same code are available due to frame header loss, copy combining is performed for the purpose of correcting errors. If the combined copies of the sequence can not recover the message, then both copies are discarded. The next transmission is the encoded sequence with rate R_1 again. Therefore, error detection is performed only at every odd transmission, and error correction is carried out at every even retransmission.

The state diagram for this scheme is shown in Figure 3.6. When the header of the frame is successful and one of the sequences with rates R_1 and R_2 is received, the sequence is checked for error. If errors are detected, the system moves to the next state labelled $\frac{R_1R_2}{R_2R_1}$

depending on whether the encoded sequence with rate R_1 is transmitted first or that with rate R_2. When the second sequence is transmitted, the header can be in error, in which case the system move to the state $\frac{R_1 R_1}{R_2 R_2}$. Otherwise, the sequence is combined with the previous copy, and if the Viterbi decoding is not successful, the sequences are discarded, and the transmission starts all over again. Moreover, if due to damaged header two copies of encoded sequences with rate R_1 and R_2 are repeated, copy combining is performed.

The expected number of transmission for a packet in the conventional Type-II ARQ is given by the upper bound

$$\begin{aligned} \bar{N} &\leq \mathrm{V}\left(I - G'\right)^{-1}\mathbf{W} \\ &\leq \frac{(1+\mathrm{h})\left(1+X_1^{(1)}\right)}{(1-\mathrm{h})\left(1-X_1^{(2)}\right)+\mathrm{h}(1-\mathrm{h})\left(1-X_2^{(2)}\right)} + \frac{\mathrm{a}}{(1-\mathrm{h})(1-\mathrm{a})} \end{aligned} \tag{3.20}$$

In the expression for $\bar{N}$, the decoding error probabilities $X_1^{(1)}$, $X_1^{(2)}$, and $X_2^{(2)}$ are defined as

$$\begin{aligned} X_1^{(1)} &= 1-(1-p)^L \\ X_1^{(2)} &\leq 1-(1-P(E))^L \\ X_2^{(2)} &= 1-(1-p_2)^L, \end{aligned} \tag{3.21}$$

where $P(E)$ is the error event probability of the Viterbi decoding which is given by Equation (2.1). p is the channel error rate and p_2 is calculated by replacing d with 2 in the Equation (2.2).

Now, we present the numerical and simulation results to verify the preceding analysis for the Type-II HARQ schemes. The packet length is 900 bits with a block code of 32 bits for error detection purposes. The code used for the Type-II HARQ scheme is a rate 1/2 Convolutional Code with memory 4. It is listed in the Table 1 along with its weight spectra.

Table 1 Distance Spectra of the Convolutional Code (2, 1, 4) and (3,1, 4)

R	Generator Polynomial	d_{free}	$(a_{d_{free+j}}, j = 0, 1, 2, ...)$ $(c_{d_{free+j}}, j = 0, 1, 2, ...)$
1/2	(23, 35)	7	(4, 12, 20, 72, 225, 500, ...) (2, 3, 4, 16, 37, 68, ...)
1/3	(23, 35)	10	(1, 0, 6, 0, 12, 0, ...) (1, 0, 27, 0, 58, 0, ...)

Table 2 Distance Spectra of Punctured Codes (4, 3, 4), (6, 3, 4), and (9, 3, 4)

R	Perforation Matrix	d_{free}	$(a_{d_{free+j}}, j = 0, 1, 2, ...)$ $(c_{d_{free+j}}, j = 0, 1, 2, ...)$
3/4	$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852 ...) (1, 7, 125, 936, 5915, 36608, ...)
3/6	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	7	(6, 9 , 12, 48, 111, 204, ...) (12, 36, 60, 216, 675, 1500, ...)
3/9 1/3	$\begin{bmatrix} 2 & 2 & 1 \\ 2 & 1 & 1 \end{bmatrix}$	10	(5, 7, 5, 12, 23, 41, ...) (16, 26, 21, 61, 127, 241, ...)

Punctured Convolutional Codes with memory 4 [28] are also used for encoding the frame header and the ACK. They are selected from Table 2, in which the Punctured Codes are obtained from an original rate 1/2 code with Generator Polynomials (23, 35).

Figure 3.7 shows the performance of Type-II HARQ scheme with Punctured Convolutional Codes of different rates for the frame header in noise-free feedback channels. The header failure rate when employing a Punctured Code with rate k/n is given by

$$\mathrm{h} \leq 1 - (1 - P(E))^{L_\mathrm{h}/\mathrm{k}}, \tag{3.22}$$

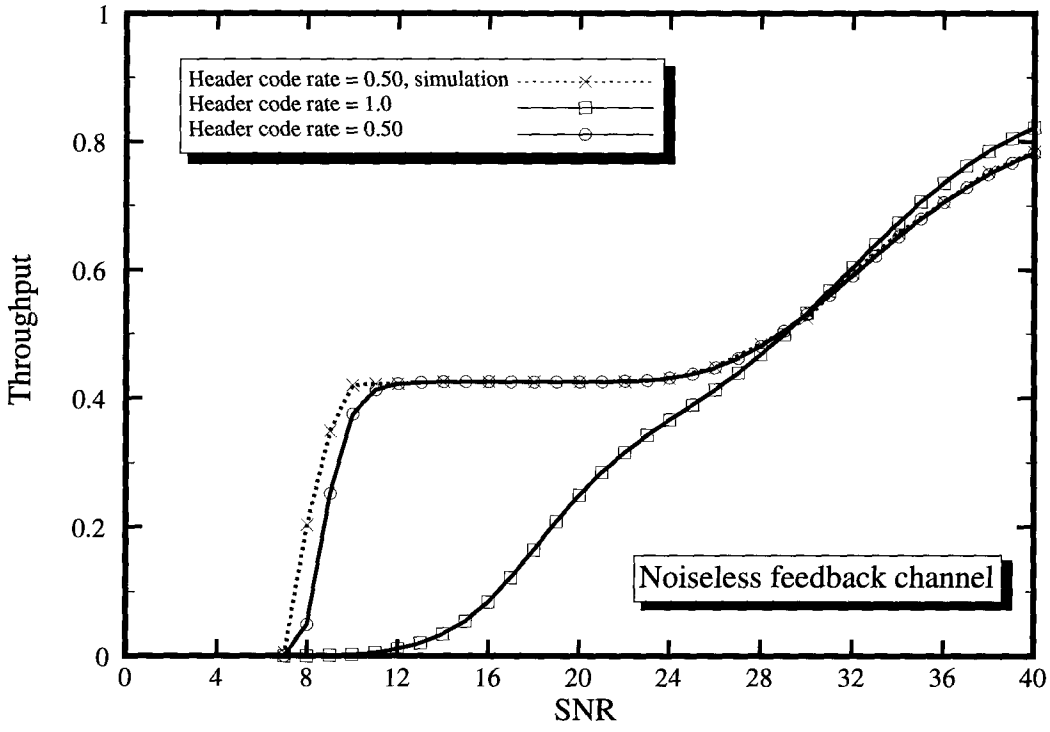

Figure 3.7 Throughput of the Conventional Type-II ARQ Scheme with the Effects of Separate Header

where *P(E)* is given by Equation (2.1). It is seen that when the code rate 1/2 is used for protection of header, the performance is remarkable. Thus, with the frame header code rate being low enough to combat the channel noise, the majority of packets reach the receiver. Although the successfully received packet might have been corrupted by the channel noise, it can be exploited to reduce the probability of uncorrectable errors when additional copies are available. On the other hand, with the code rate 1 for the header, the performance is not satisfactory, and there is sharp drop in throughput at moderate and low SNR values. In this case, the amount of repetition redundancy with error correction at the receiver is substantially reduced. The consequence is an increase in the average number of transmissions.

Figure 3.8 show that the efficiency of the system falls drastically in the Type-II ARQ

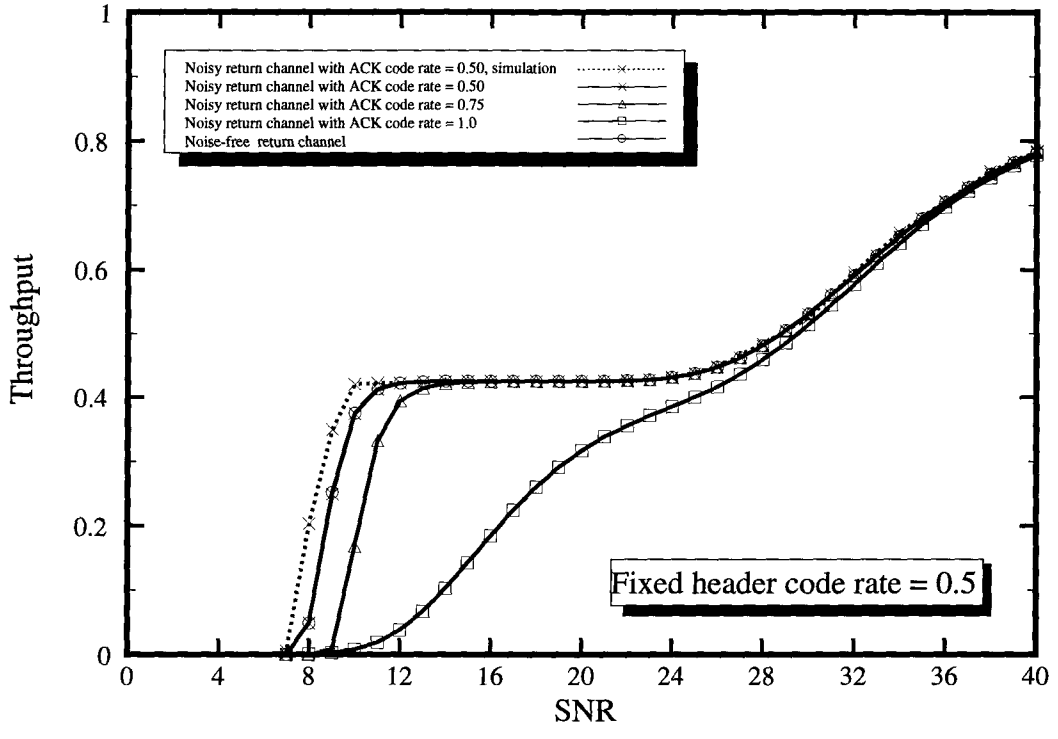

Figure 3.8 Throughput of Type-II with Noisy Feedback Channel

scheme if the ACK is not protected adequately against the noisy return channel. Very poor performance is resulted when the frame header encoded with rate 1/2 but the ACK is sent back without enough error-correction protection. The amount of the degradation is as high as 400% at 10dB SNR. As more powerful codes are employed , this deterioration decreases, such is the case with rate 1/2 convolutional code for the ACK.

3.2.3 Generalized Self-Decodable (GSD) Type-II ARQ Scheme with CPC Codes

The GSD Type-II ARQ scheme employs code combining on the received sequences encoded with equivalent codes to improve the performance of the system [4]. At high SNR a fixed high-rate code is used to combat the nominal channel noise, whereas at low SNR

the equivalent codes are combined to produce a low-rate code with enough error-correcting capability to recover the message.

The General Scheme becomes equivalent to the GSD Type-II HARQ if the parameter R_t is a high-rate Punctured Convolutional Code, and t = z, where z represents the number of equivalent codes which can exist for the high-rate code derived from an original rate. Following the assumptions made for the General scheme, the Markov chain for this transmission strategy with t = 3 will have 15 states, two of which are the sink and ACK states respectively.

Three equivalent codes FEC (4, 3, 4) are used and repeated based on the channel requirement. These codes along with other CPC Codes used to obtain the numerical and simulation results are included in Table 3. Combining the equivalent codes in level 2 of the algorithm lead to new codes which have the same distance spectra. Therefore, the following relationship holds:

$$X_1^{(2)} = X_2^{(2)}. \tag{3.23}$$

The same simplification is applicable to the resulting codes in level 3. Then, we have

$$X_1^{(3)} = X_2^{(3)} = X_3^{(3)} = X_4^{(3)} = X_5^{(3)} = X_6^{(3)} = X_7^{(3)} = X_8^{(3)} \tag{3.24}$$

The different decoding failure probabilities for the codes with rates k/n are obtained from

$$\begin{aligned}
X_1^{(1)} &\leq 1 - \left(1 - [P(E)]_{C_1}\right)^{L/\mathrm{k}}, \\
X_1^{(2)} &\leq 1 - \left(1 - [P(E)]_{C_1+C_2}\right)^{L/\mathrm{k}}, \\
X_3^{(2)} &= 1 - (1 - p_2)^L, \\
X_1^{(3)} &\leq 1 - \left(1 - [P(E)]_{C_1+C_2+C_3}\right)^{L/\mathrm{k}}, \\
X_9^{(3)} &= 1 - (1 - p_3)^L,
\end{aligned} \tag{3.25}$$

Table 3 Distance Spectra of CPC Codes of Memory 4 with rate 3/4

R	C_i	Perforation Matrix	d_{free}	$(a_{d_{free}+j}, j = 0,1,2,...)$ $(c_{d_{free}+j}, j = 0,1,2,...)$
3/4	C_1	$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/4	C_2	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/4	C_3	$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/8	C_1+C_2	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	8	(1, 4, 3, 11, 18, 38, ...) (1, 11, 11, 45, 86, 204, ...)
3/8	C_2+C_3	$\begin{bmatrix} 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	8	(1, 4, 3, 11, 18, 38, ...) (1, 11, 11, 45, 86, 204, ...)
3/8	C_1+C_3	$\begin{bmatrix} 1 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix}$	8	(1, 4, 3, 11, 18, 38, ...) (1, 11, 11, 45, 86, 204, ...)
3/12	C_1+C_2+C_3	$\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}$	14	(6, 0, 9, 0, 12, 0, ...) (12, 0, 36, 0, 60, 0, ...)

where the subscript of $P(E)$ indicates decoding error event when the corresponding CPC Code is used. p_2 and p_3 are obtained from Equation (2.2) with $d = 2$ and $d = 3$.

The performance curves for the Generalized Type-II HARQ scheme with CPC codes are shown in Figure 3.9. Similar to the previous ARQ strategies, it is seen that, in this scheme, the performance improvement is significant when the header is reasonably protected against the channel noise . In other words, high success rate of header increases the chances of code combining operation for equivalent codes. For instance, the system performance can

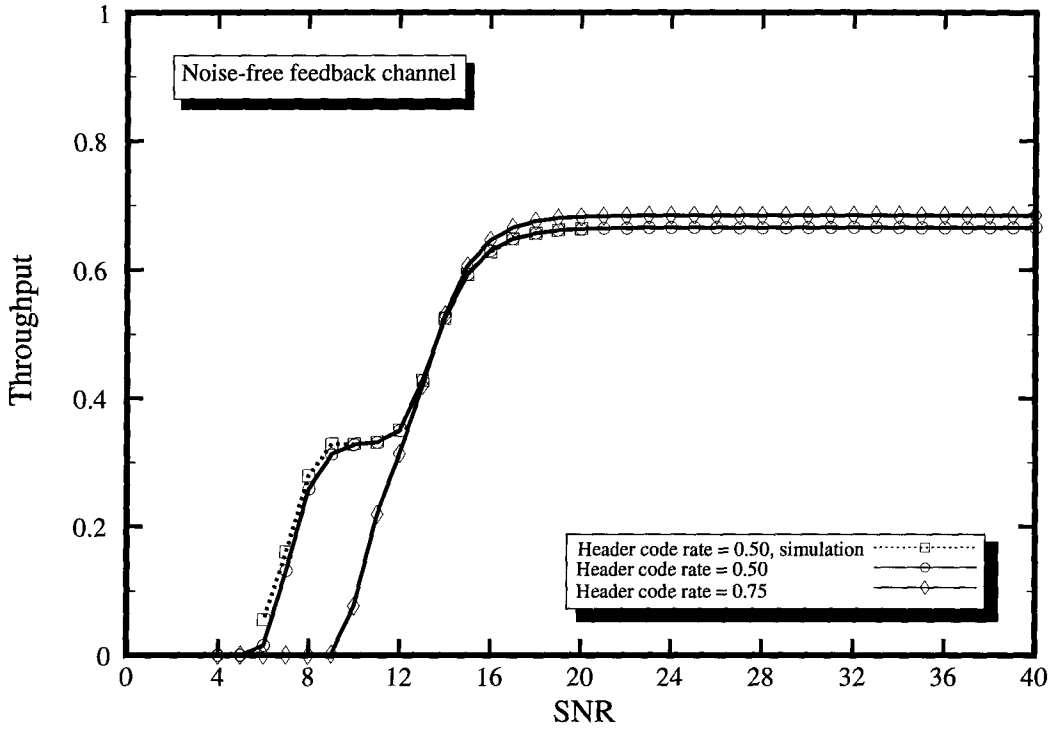

Figure 3.9 Throughput of GSD Type-II ARQ Scheme with Three Equivalent Codes

be improved by 200% if the header code rate 0.5 is used instead of the code with rate 0.75

The noisy feedback channel also plays a major role in the deterioration of the system performance. In Figure 3.10, the header of the frame in the forward channel is encoded with a fixed rate 1/3, whereas the ACK is protected with codes of different rates. Similar to the case of header, enough error protection is required for the ACK to obtain satisfactory system performance. At 10 dB and 20 dB the performance improvements are remarkable when the ACK is encoded with the rate 1/2. The improvement ranges form 67% at 20dB to 200% at 10dB compared to the uncoded ACK case.

3.3 Optimization of Header and ACK Sensitivities

As pointed out, errors in the feedback channel and header failure reduce the throughput

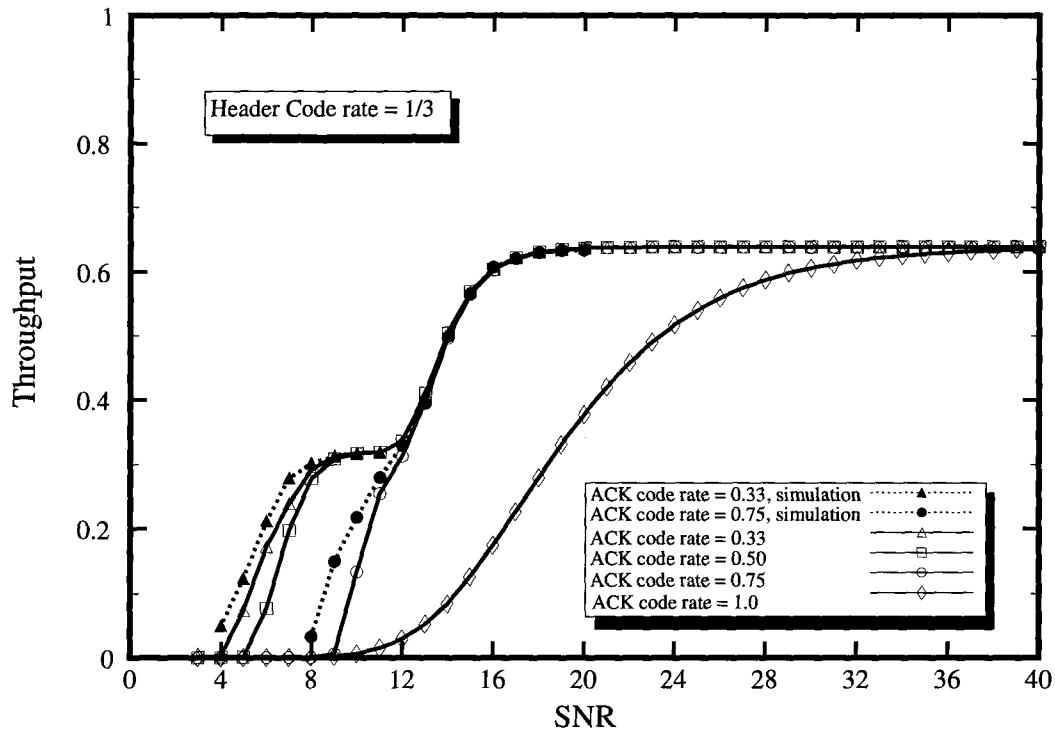

Figure 3.10 Throughput of GSD Type-II ARQ Scheme with Three Equivalent Codes and Noisy Feedback Channel

performance by lengthening the time during which the transmitter is uncertain of the outcome of a transmitted packet. These factors will also cause wasted transmissions if the data packet is lost because of header failure or if the given packet has already been successfully received unknown to the transmitter. Moreover, the advantage offered by MARQ schemes can no longer be obtainable when frequent packet loss occurs due to insufficiently protected frame header.

The effects of these factors were also investigated with numerical and simulation results. It was discovered that when the same code rate or smaller as that for successful decoding of the data field was used for header, the desirable performance could be achievable. This step ensures that the amount of employed overhead is optimum at each transmission. For

instance, when the required data rate for successful decoding was 3/4, the header code rate should have been 3/4 too. When Viterbi decoding with rate 1/2 was performed to recover the message, the same error-correcting code should also have been used for encoding the header.

Therefore, to minimize the influence of these parameters on the throughput, it is recommended that the employed code for protection of the frame header should possess the same error-correcting capability as that used for the successful decoding of the data packet. This measure ensures that the header failure rate is kept well below the error rate of the data field after error correction since the header length is much shorter than the data field.

The same conclusions are applicable to the case of imperfect feedback channel because in most cases the ACK is included in the frame header in the return channel , where the frame has the same structure as that in the forward channel.

Chapter 4 Performance of Adaptive Hybrid ARQ Schemes in the Time-Varying Channels

ARQ schemes with fixed FEC codes perform reasonably well in the AWGN environment. However, they fail to yield satisfactory performance in the time-varying channels. With a fixed low-rate code, the throughput can suffer during the good channel conditions. In contrast, a code with high rate may not entail sufficient error correction capability to combat a highly degraded channel. An optimum solution is to take advantage of the periods of good channel conditions and transmit with a high code rate, whereas during the noisy channel conditions the code rate must be lowered to the extent that it matches the prevailing channel conditions.

Adaptive schemes have been investigated to be effective methods for combating the time-varying nature of the distortion in the fading channels. A series of published papers report considerable gains by adapting certain parameters such as error control codes and transmission rate in the transmitter according to the channel conditions [29], [30]. In this chapter, we present adaptive ARQ systems which select the optimal code rate according to the channel state information being supplied by a channel estimator. The adaptive schemes, employing Punctured Convolutional Codes with Viterbi decoding, require one encoder/decoder for all the codes derived from an original low-rate code. Therefore, improvement in the system performance is obtained while the system complexity is kept at a minimum level.

4.1 A Type-I Adaptive SW ARQ

The adaptive ARQ scheme in this thesis is conceptually simple to implement. It is similar to that presented by Schacham [8], but the proposed system does not move to a

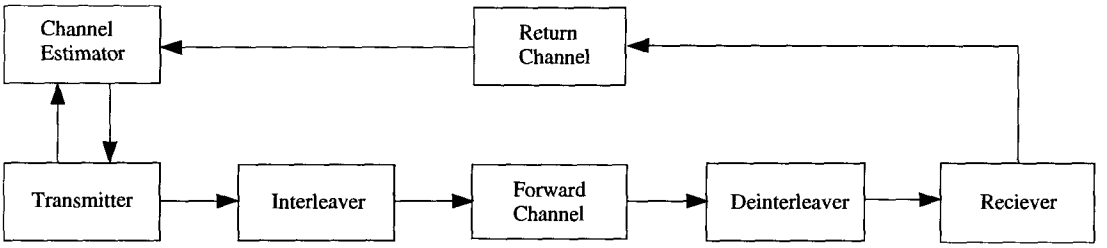

Figure 4.1 Block Diagram of the Adaptive Protocol

worse state immediately with the arrival of A NACK. Instead, it remains in the present state until the number of erroneous packets equals or exceeds a predetermined threshold. These thresholds depend on the states, and they are chosen in such a way that optimum system performance can be obtained. Moreover, unlike the Schacham's scheme which uses one observation interval, two observation intervals are adopted in our algorithm: one long interval for transition to a better state, and a short subinterval for moving to a worse state. The long interval is a multiple factor of the subinterval, and it must be selected in such way to yield an accurate estimate of the channel conditions. The subinterval is short to prevent long decision delays when the channel conditions are not good. We refer to this system as the Adaptive Type I Hybrid ARQ (AHARQ) scheme.

A simplified block diagram of the proposed scheme is shown in Figure 4.1. The data sequence of length k originating from the source is appended with n_p CRC bits for error detection purposes and m known tail bits for clearing the encoder memory. The error detecting bits are fixed, and they are chosen long enough to make the probability of undetected error very small. The resulting sequence is further encoded with one of the M codes of different rates as described in [8]. The interleaving operation is performed on the sequence, and the data is transmitted over the forward channel. At the receiving side, the

streams of data are de-interleaved and decoded. The CRC bits, then, are used to determine the existence of errors in the sequence. Should errors exist, the packet is discarded and a NACK is sent back to the transmitter. We assume that the return channel is noise-free, and the propagation delay is negligible.

The transmitter decides which one of the M codes should be used at the start of every transmission according to a set of given thresholds that are chosen to keep the throughput above a certain level at each SNR value. We assume there are M possible code rates, $r_1 > r_2 > ... > r_M$, to choose from. The code rate r_M is used during the worst and the rate r_1 during the best channel conditions. Since there are M different codes being employed, we also have M different adaptation thresholds, n_1, ..., n_M.

While being in state i, or while using rate r_i, the transmitter counts the number of erroneous packets, n_e, which are transmitted during an observation interval, N, where N is expressed in packets. If $n_e \geq n_i$, the next lower rate r_{i+1} should be used. On the other hand, if $n_e < n_i$ in each interval N for α_i consecutive observation intervals, the higher rate r_{i-1} is selected. The above algorithm can be summarized as follows:

1. Transmit at rate r_1. If there are n_1 packets in error in N transmitted packets, switch to rate r_2.
2. Transmit at rate r_i, $1 < i < M$, as long as the number of errors is less than n_i in an observation interval N. Should n_i errors occur, switch to r_{i+1}. In contrast, if there are less than n_i errors in N packets for α_i consecutive observation intervals, switch to rate r_{i-1}.
3. Transmit at a code rate r_M as long as there are more than n_i errors in N packets. If there are less than n_i errors in N packets for α_i consecutive observation intervals, then

switch to rate r_{M-1}.

Optimum values must be determined for the system parameters N, n_i and α_i to maximize the throughput performance. N and n_i are selected in such a way that the relationship

$$\frac{N - n_i}{N} r_i \geq r_{i+1} \tag{4.1}$$

is satisfied all the time when the system is in state i. In this expression, the equation, $\frac{N-n_i}{N} r_i$, provides the transmitter with the estimate of the throughput while the rate r_i is being used. Having chosen suitable values for the threshold parameters n_i and N, we select α_i to meet the channel requirements. Small α_i is desirable to avoid long decision delays at good channel conditions, whereas large α_i must be chosen to yield accurate channel estimation in degraded channels.

It should be noted that due to the assumption of noise-free feedback channel, the receiver is assumed to be aware of the instants of the rate change. Moreover, perfect frame synchronization is assumed in the analysis of throughput of the Adaptive Type-I HARQ. To conform this system to the practical case, the information about the code rate could be included at the beginning of the packet. This information would be encoded with the lowest available code rate, i.e. r_M, which would be known to the receiver. The introduced overhead would affect the throughput efficiency, but the degradation incurred would be negligible. This latter scenario is investigated in section 2 where the code rates for both the data and frame header are adapted according to the channel conditions.

4.1.1 Throughput Analysis

The throughput of this system can be found by modeling the time-varying channels as

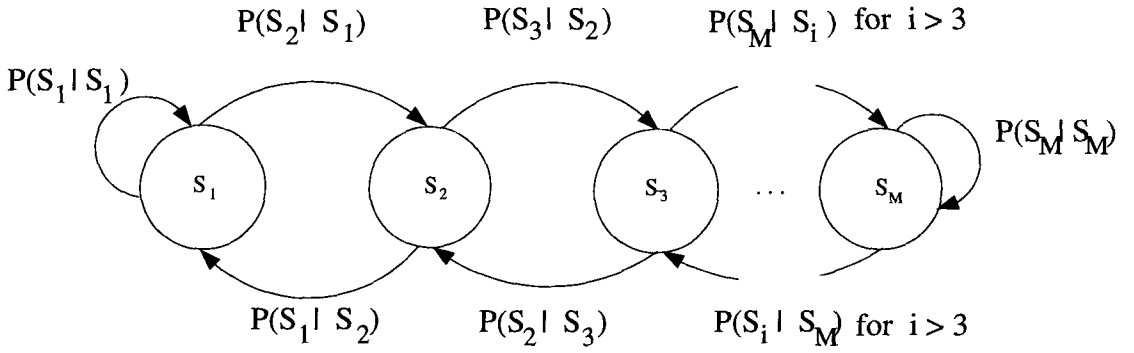

Figure 4.2 State Transition Diagram for the Adaptive Scheme

consisting of M stationary BSC channels [8]. This model is based on the assumption that the channel parameters are varying slowly. Consequently, only the transitions to the two adjacent states are allowed where the states are ordered according to the severity of channel conditions. Also, the transition from the state S_k to the next state not only depends on the present state, but also on the length of the time that has been spent in the present state. This is, therefore, a discrete-time Semi-Markov process [31] with M possible states, in which the amount of time spent in a given state is a discrete random process.

The state diagram of the proposed scheme is shown in figure 4.2. The states are ordered according to the decreasing code rates; state 1 is used to represent the best condition of the channel, whereas state M models the worst case of the channel degradation. The transition probabilities depend on the length of time the semi-Markov process spends in a state. The predetermined thresholds for erroneous packets and the observation intervals determine the instants at which the transitions should occur.

Let X denote the number of packets that are detected in error. The transition form state S_i to S_{i-1} will occur if the number of erroneous packets in each interval N is less than n_i for

α_i consecutive observation intervals. The probability of such an event is given by

$$P(S_{i-1} \mid S_i) = [P(X < n_i)]^{\alpha_i}, \qquad 1 < i \leq M \tag{4.2}$$

where

$$P(X < n_i) = \sum_{j=0}^{n_i-1} \binom{N}{j} p_{e_i}^j (1 - p_{e_i})^{N-j}. \tag{4.3}$$

In equation (4.3), p_{e_i} is the probability that a packet has to be retransmitted while being in state i. This probability is defined as

$$p_{e_i} = 1 - (1 - P(E_i))^L, \tag{4.4}$$

where L is the length of the packet and $P(E_i)$ is the error event probability for the error correction code used in state i.

If there are n_i errors in one of the N packets transmitted, then the transition to a lower state is necessary to achieve optimal performance. This transition probability is expressed as

$$P(S_{i+1} \mid S_i) = 1 - [P(X < n_i)]^{\alpha_i}, \qquad 1 \leq i < M. \tag{4.5}$$

There is no transition from state 1 to a better state or from state M to a worse state, and these transition probabilities, $P(S_1 \mid S_1)$ and $P(S_M \mid S_M)$ are found as follows:

$$\begin{aligned} P(S_1 \mid S_1) &= [P(X < n_1)]^{\alpha_i} \\ P(S_M \mid S_M) &= 1 - [P(X < n_M)]^{\alpha_i} \end{aligned} \tag{4.6}$$

The steady state probabilities are defined as the row matrix $[W] = [W_1, W_2, ..., W_M]$, and they are the solutions of the equation $[W][P] = [W]$, where P is the transition matrix whose components are given by the transition probabilities (4.2), (4.5), and (4.6).

To calculate the throughput, we need to know the expected total number of packets transmitted and the expected number of successful packets (while staying in the i-th state) before making a transition to either a better or a worse state.

Assume that $n_i + k$ packets have been sent, out of which n_i packets are in error, the probability of such transmissions is given by

$$\begin{aligned} P\{n_i + k\} &= p_{ei} C_{n_i-1}^{n_i-1+k} p_{ei}^{n_i-1} (1 - p_{ei})^k, \qquad k = 0, 1, 2, ..., N - n_i \\ &= \binom{n_i - 1 + k}{n_i - 1} p_{ei}^{n_i} (1 - p_{ei})^k. \end{aligned} \tag{4.7}$$

The first p_{ei} indicates that the latest packet has to be in error for a transition to a lower state to occur. $p_{ei}^{n_i-1}$ is the probability that $n_i - 1$ packets were in error before this latest erroneous packet, and $\binom{n_i-1+k}{n_i-1}$ is the binomial coefficient given by

$$\binom{n_i - 1 + k}{n_i - 1} = \frac{(n_i - 1 + k)!}{(n_i - 1)! k!}. \tag{4.8}$$

If the transition does not take place because of having less than n_i error in the first interval N, the system is in the second interval $2N$. The probability that $N + n_i + k$ packets are transmitted before a transition to a lower state is given by

$$P\{N + n_i + k\} = P(x < n_i) P(n_i + k) \tag{4.9}$$

The same argument applies to α_i observation intervals. In other words,

$$P\{\gamma N + n_i + k\} = [P(x < n_i)]^\gamma P(n_i + k), \quad \gamma = 1, 2, ..., \alpha_i - 1 \tag{4.10}$$

If there are less than n_i errors in each N transmitted packets for $\alpha_i N$ consecutive transmissions, the system moves to a better state. The probability of such an event is expressed by

$$P\{\alpha_i N\} = [P(x < n_i)]^{\alpha_i} \tag{4.11}$$

Therefore, the expected number of packets transmitted during a stay in a state before a transitions to the next state can be put into the closed form

$$E[T_i] = \sum_{j=0}^{\alpha_i - 1} \sum_{k=0}^{N-n_i} (jN + n_i + k) \binom{n_i - 1 + k}{n_i - 1} p_e^{n_i} (1 - p_e)^k [P(x < n_i)]^j + \alpha_i N [P(x < n_i)]^{\alpha_i}, \tag{4.12}$$

and the expected number of packets the process stays between transitions is obtained by

$$E[T] = \sum_{i=1}^{M} W_i E[T_i]. \tag{4.13}$$

As indicated earlier, the expected number of data transmitted successfully during a stay in state i is also needed for the throughput calculation. Assuming $\alpha_i = 3$, and let k be the number of correct packets. In the first observation interval, a transition to a lower state occurs if n_i packets are detected in error. The expected number of correct packets in the first N interval is

$$\sum_{k=0}^{N-n_i} (k) \binom{n_i - 1 + k}{n_i - 1} p_{ei}^{n_i} (1 - p_{ei})^k \tag{4.14}$$

where the number of correct packets can vary form 0 to $N - n_i$.

However, if the number of packets in error in the first observation interval is less than n_i, then there is at least $N - n_i + 1$ correct packets among the first N transmitted packets. There could be another 0 to $N - n_i$ correct packets in the first and second N, $n_i - 1$ of which could be in the first N transmitted packets. Then, the expression for the expected number of successful packets transmitted during 2 observation intervals before a transition to the

lower state is given by

$$\sum_{k=0}^{N-n_i}(N-n_i+1+k)\sum_{i_1=0}^{min(k,n_i-1)}\binom{N}{N-n_i+1+i_1}(1-p_{e_i})^{N-n_i+1+i_1} \\ p_{ei}^{n_i-1-i_1}\binom{n_i-1+k-i_1}{n_i-1}p_{e_i}^{n_i}(1-p_{e_i})^{k-i_1} \tag{4.15}$$

The restriction **min** on the summation sign is due to the fact that there could be only $n_i - 1$ extra correct packets in the first interval. This expression takes into account the different possibilities of the correct packets occurring in the interval $2N$ provided we did not switch to a worse state during the first N transmitted packets.

The condition for the system to remain in a state after $2N$ packets have been transmitted depends on the number of erroneous packets in each of the interval N. Thus, to send the next packet in this state, we must have had less than n_i errors in each of the N previously transmitted packets. The number of correct packets is $2(N - n_i + 1) + k$, out of which $2(N - n_i + 1)$ are among the $2N$ transmitted packets. While the system is in this state, there could also be another 0 to N-n_i correct packets; $n_i - 1$ of them could be in the first interval and another $n_i - 1$ in the second interval. The expected number of successful packets sent during the $3N$ transmitted packets before a transition to a lower state is expressed by

$$\sum_{k=0}^{N-n_i}(2(N-n_i+1)+k)\sum_{i_1=0}^{min(k,n_i-1)}\binom{N}{N-n_i+1+i_1}(1-p_{e_i})^{N-n_i+1+i_1} \\ p_{ei}^{n_i-1-i_1}\sum_{i_2=0}^{min(k-i_1,n_i-1)}\binom{N}{N-n_i+1+i_2}(1-p_{e_i})^{N-n_i+1+i_2}p_{ei}^{n_i-1-i_2} \\ \binom{n_i-1+k-i_1-i_2}{n_i-1}p_{e_i}^{n_i}(1-p_{e_i})^{k-i_1-i_2}. \tag{4.16}$$

The successfully transmitted packets before the transition to a higher state must also be calculated. This scenario occurs when there are less than n_i errors in the first N packets,

less than n_i errors in the second N packets, and less than n_i errors in the third N transmitted packets. Thus, the expected number of successful packets is greater or equal to $3(N - n_i + 1)$, and is obtained from

$$\sum_{k=0}^{2(n_i-1)} (3(N-n_i+1)+k) \sum_{i_1=0}^{min(k,n_i-1)} \binom{N}{N-n_i+1+i_1} (1-p_{ei})^{N-n_i+1+i_1} p_{ei}^{n_i-1-i_1} \sum_{i_2=0}^{min(k-i_1,n_i-1)} \binom{N}{N-n_i+1+i_2} (1-p_{ei})^{N-n_i+1+i_2} p_{ei}^{n_i-1-i_2} \binom{N}{N-n_i+1+A} (1-p_{ei})^{N-n_i+1+A} p_{ei}^{n_i-1-A} \tag{4.17}$$

where $A = k - i_1 - i_2$. This term will be zero if $A > n - 1$ due to the fact that some combinations do not occur. The total number of packets successfully transmitted, $E[S_i]$, during a stay in state i with $\alpha_i = 3$ is the sum of the Equations (4.14), (4.15), (4.16), and (4.17). This discussion can be extended further to greater values of α_i which involves having more terms in the expression for expected number of successful packets transmitted in a given state i.

The throughput of the protocol is then given by

$$\eta = \frac{\sum_{i=1}^{M} W_i r_i E[S_i]}{\sum_{i=1}^{M} W_i E[T_i]}, \tag{4.18}$$

where r_i is the code rate used in state i.

4.1.2 Numerical Analysis

To illustrate the effectiveness of the proposed adaptive scheme, we select the case in which the channel model is represented by four states. The code with rate 1 is selected for state 1, whereas the FEC (8, 7, 4), (4, 3, 4), and (2, 1, 4) Punctured Convolutional Codes

Table 4 Distance Spectra of Punctured Codes (8, 7, 4) , (4, 3, 4), and (2, 1, 4)

R	Perforation Matrix	d_{free}	$(a_{d_{free}+j}, j = 0,1,2,...)$ $(c_{d_{free}+j}, j = 0,1,2,...)$
7/8	$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$	3	(13, 145, 1471, 14473, 143110, 1416407, ...) (49, 1414, 21358, 284324, 3544716, 42278392, ...)
3/4	$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/6 1/2	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	7	(6, 9, 12, 48, 111, 204, ...) (12, 36, 60, 216, 675, 1500, ...)

are employed in the states 2, 3, and 4 respectively. These codes are listed in Table 4. The number of information bits is 900 with 32 CRC bits appended at the end for error detection purposes. The probability that the packet encoded with a code having rate 1 must be retransmitted is calculated from

$$p_{e_1} = 1 - (1-p)^L, \tag{4.19}$$

in which p is the channel error rate and L is the packet length. The probability that decoded message contains errors while it was encoded with a rate r_i= k/n is given by

$$p_{e_i} \leq 1 - \left(1 - P(E)_{r_i}\right)^{L/\mathrm{k}}, \qquad i = 2,3,4 \tag{4.20}$$

where *P(E)* is the error event probability of Viterbi decoder given by Equation (2.1).

The Expression (4.1) can be rearranged to obtain

$$N \geq \frac{n_i r_i}{r_i - r_{i+1}}, \tag{4.21}$$

in which there is a direct relationship between N and n_i. Therefore, the minimum value of n_i (i.e n_i = *1*) enables us to find the minimum value of N which is desirable to avoid estimation

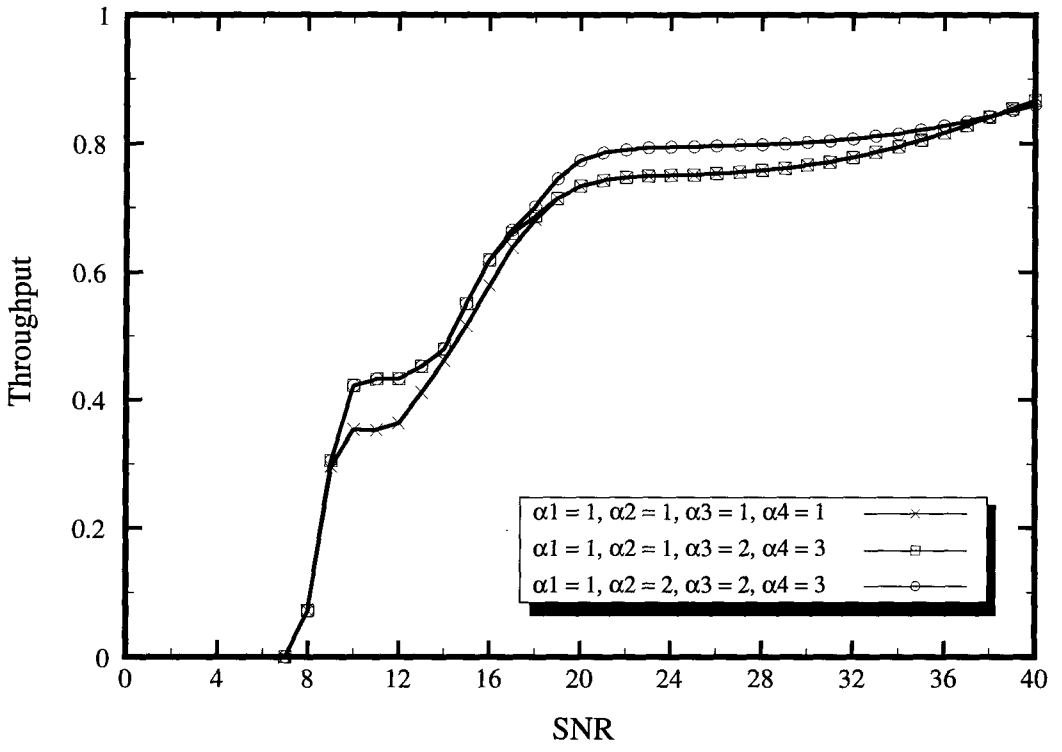

Figure 4.3 Throughput of the AHARQ with Variable α_i

delays. With $n_i = 1$, we get $N \geq 8$, $N \geq 7$, $N \geq 3$, $N \geq 1$, from which we chose the value 8 which satisfies the above relationship for all the states. Then, knowing N, we determine approximate values for n_i using the relationship

$$n_i = \frac{N(r_i - r_{i+1})}{r_i}, \tag{4.22}$$

and this leads to the values of $n_1 = 1$, $n_2 \simeq 2$, $n_3 \simeq 3$, $n_4 = 1$. As for the parameter α_i, it is determined by varying its value in such a way that the best throughput performance can be achieved over the whole desired SNR ranges. The numerical results are shown in Figure 4.3 for different values of α_i. The relative cost in throughput due to the effect of α_i can be kept at a minimum level if long observation intervals are selected at small and moderate SNR values and short intervals are chosen at large SNR values. It is seen that the scheme with α_1

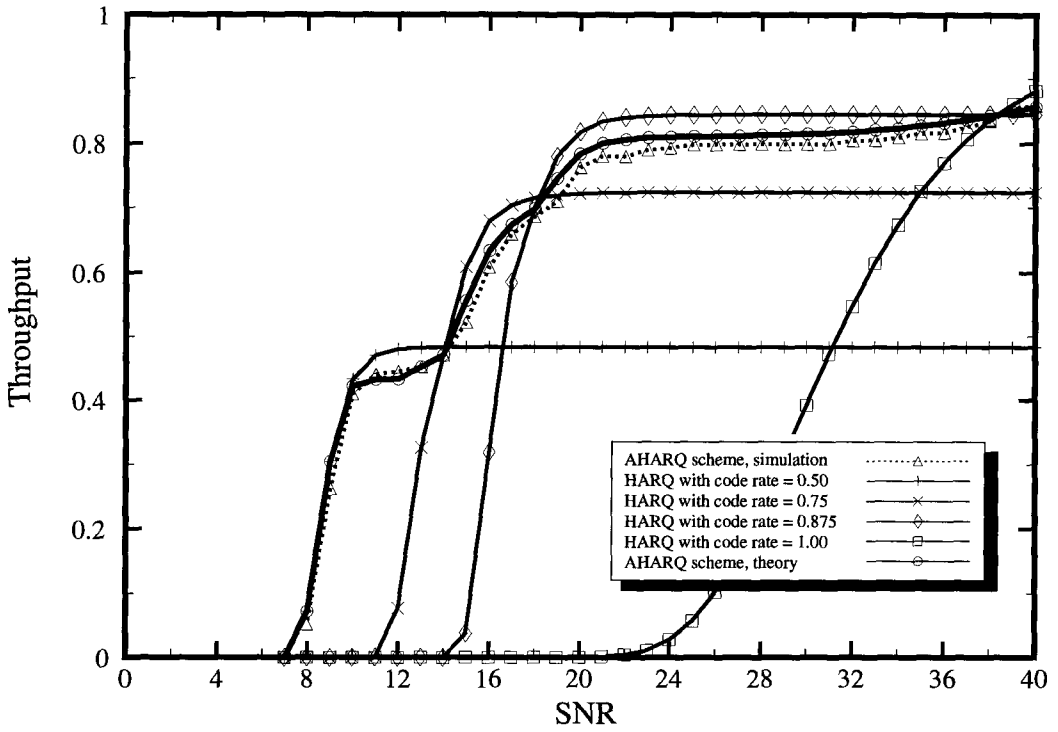

Figure 4.4 Throughput of the AHARQ and Non-Adaptive ARQ Schemes.

= 1, α_2 = 2, α_3 = 2, α_4 = 3 yields the best system performance. Increasing the value of α_i beyond 3 is discovered to not yield any further improvement in the performance. Therefore, throughout the rest of this chapter, these values of α_i are assumed unless otherwise stated.

The advantage of the AHARQ scheme compared to the non-adaptive ARQ ones is illustrated in Figure 4.4. The throughput curves show that the AHARQ scheme performs remarkably well and performance improvement of up to 700% or more can be achieved with the minimum system complexity. Simulation results in Figure 4.4 also confirm the validity of the algorithm.

The analysis presented above will be simplified to that in [8] if the n_i and α_i are set to 1 in the derived Equations (4.2 – 4.17). Figure 4.5 shows the theoretical difference in

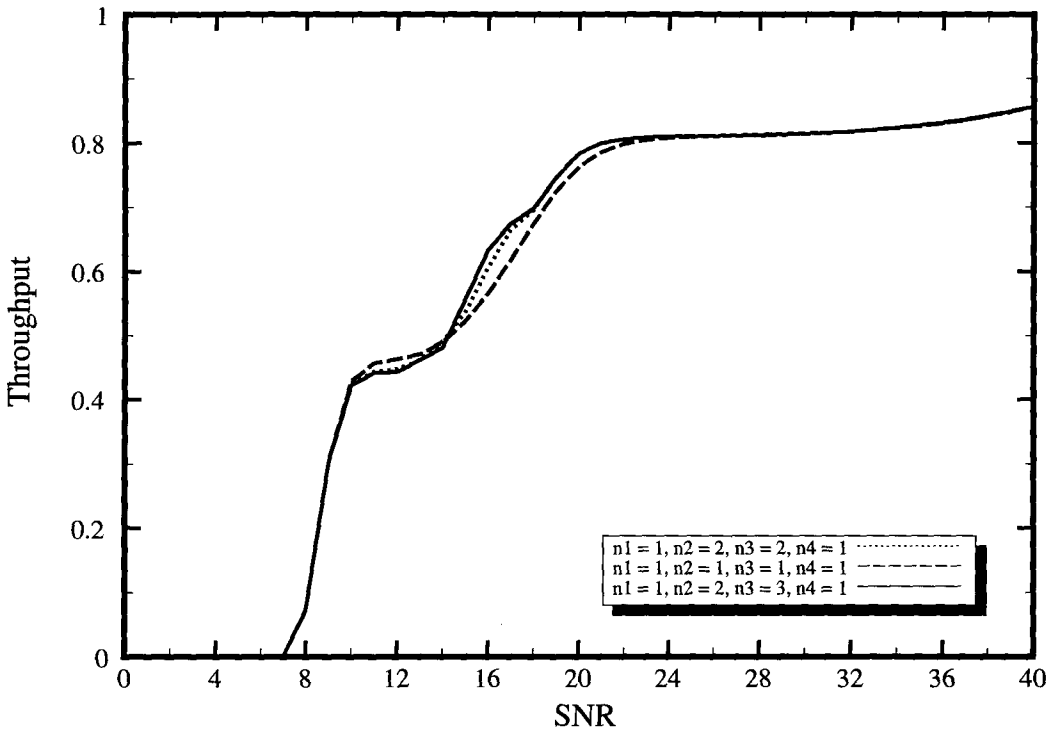

Figure 4.5 Throughput of the System with Different Threshold, n_i

performance when $n_i = 1$ compared to the case when n_i's depend on the state i. Although the ideally interleaved scenario is considered, the throughput improvement is still noticeable with n_i being dependent on state i because small n_i's lead to estimation errors. The performance improvement ranges from 2% to 9%, but more improvement can be obtained in the finite interleaving situations as will be shown later.

4.1.2.1 Effects of Interleaving and Vehicle Speed

Interleaving make the burst error patterns look like the purely random error patterns. Conventional interleaving and deinterleaving consists of reading encoded symbols into the row of a matrix. Once full, the contents of the matrix are read out by columns. The interleaving matrix has dimensions $d \times s$, where d is the interleaving depth and s the

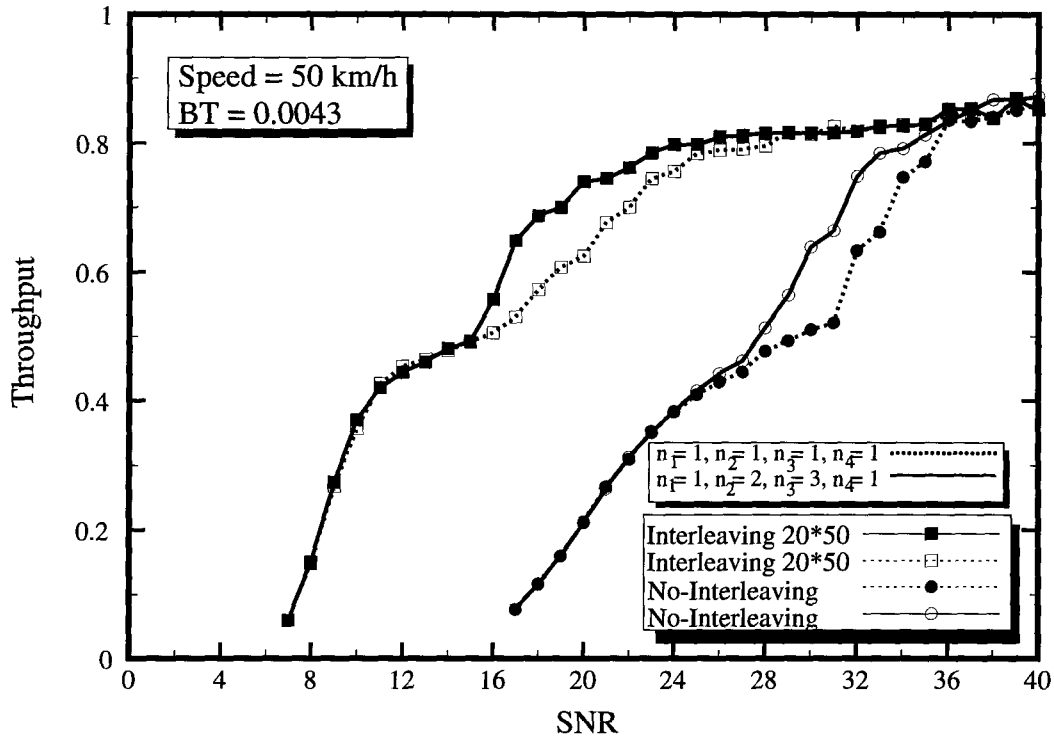

Figure 4.6 Throughput Comparison of the Adaptive Scheme for Different Values of n_i and Speed = 50 km/h.

interleaving span. At the receiver, the symbols are reconstructed by a deinterleaver identical to the original interleaver. If the interleaver has sufficient depth, the noise and fading process affecting adjacent symbols will be uncorrelated.

Unlike the ideal interleaving case in which the BER is constant, the effectiveness and superiority of the proposed algorithm is quite apparent with finite interleaving situations. The simulation results in Figures 4.6 and 4.7 illustrate the comparative throughput performance of the proposed algorithm and that of Shacham [8] with different interleaving depths. In the plots, the dotted line is used to represent the case $n_i = 1$ (i.e. Schacham's algorithm), and solid line is used for representation of the case of $n_1 = 1$, $n_2 = 2$, $n_3 = 3$, $n_4 = 1$. It is evident that interleaving operations result in considerable improvement in the performance. The

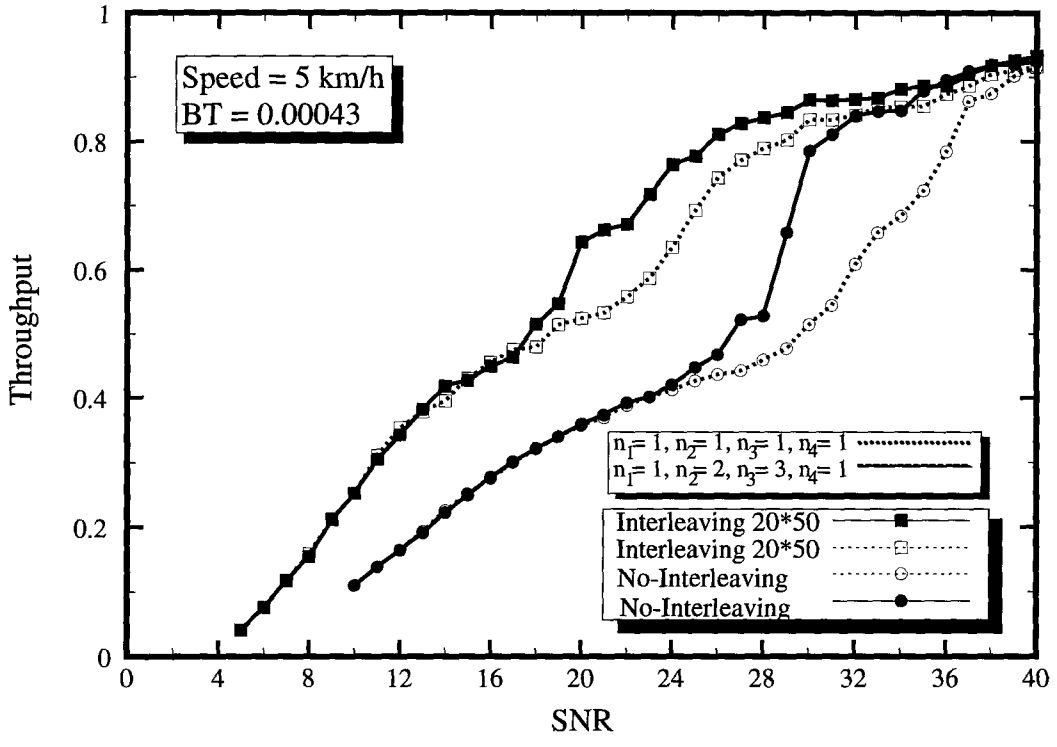

Figure 4.7 Throughput Comparison of the Adaptive Schemes for Different Values of n_i and Speed of 5 km/h.

interleaving operation removes the correlation between adjacent symbol provided sufficient interleaving depth is used. As the interleaving depth is increased, the correlation between the adjacent symbol is reduced until any further increase would not result in any more improvement. The interleaving depth of 20 proved to be nearly optimum for these specific cases. Interleaving of greater depths is required for the very bursty channels such is the case in Figure 4.7. It is demonstrated that the proposed scheme always perform better than that of Schacham [8] due to minimization of channel estimation errors with optimum values of n_i.

At low speeds and high SNR values, the noise occurs in bursts, and errors are, therefore, concentrated in fewer packets. In contrast, at low and moderate values of SNR, the random noise also contributes to the channel degradations, resulting in poor performance. As the

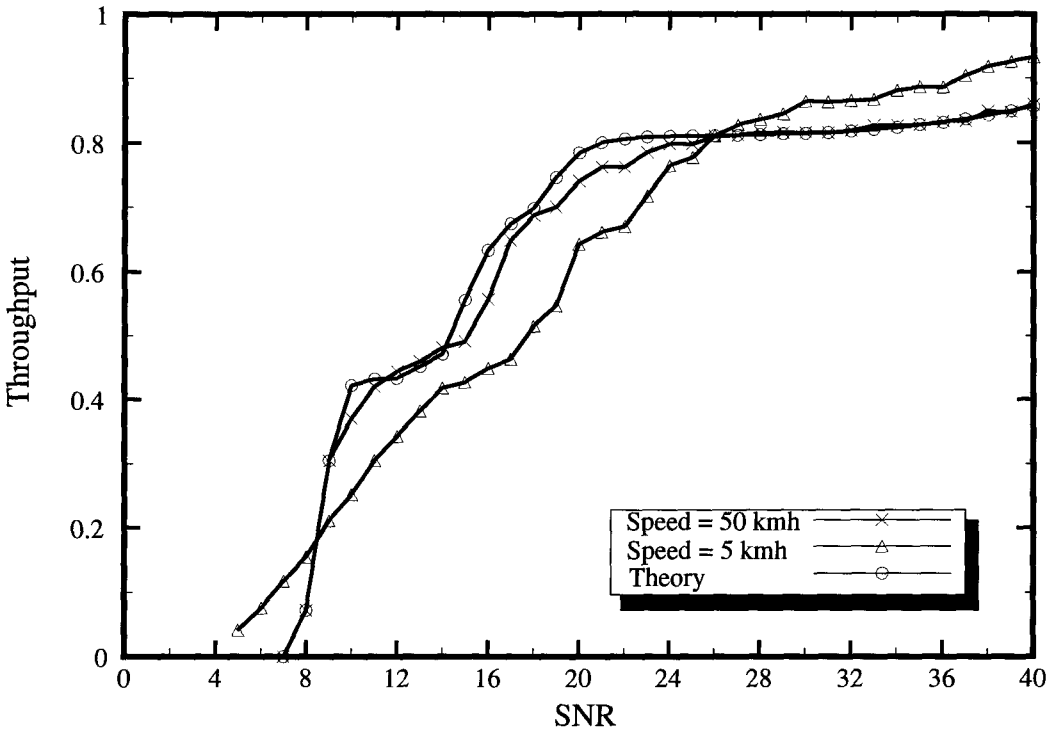

Figure 4.8 Throughput of the AHARQ at Different Speeds with Interleaving Depth of 20

speed of the vehicle increases, the fading process becomes faster and attenuated signals become less and less correlated. As a result, the system performance improves significantly. The limiting case for this improvement is the case of ideally interleaved channel where the fading elements are random. Figure 4.8 shows the throughput curves at different speeds with interleaving depth of 20. It is seen that the envelope of the throughput curve of the scheme at 50 km/h with sufficient interleaving follows closely that of the ideal interleaving case.

4.1.2.2 Effects of Packet length

In the analysis, we assumed the packet size varies with each new code rate to accommodate the required parity bits. It is also possible to keep the packet size constant and vary the information bits fed to the encoder. The comparative performance of these two cases

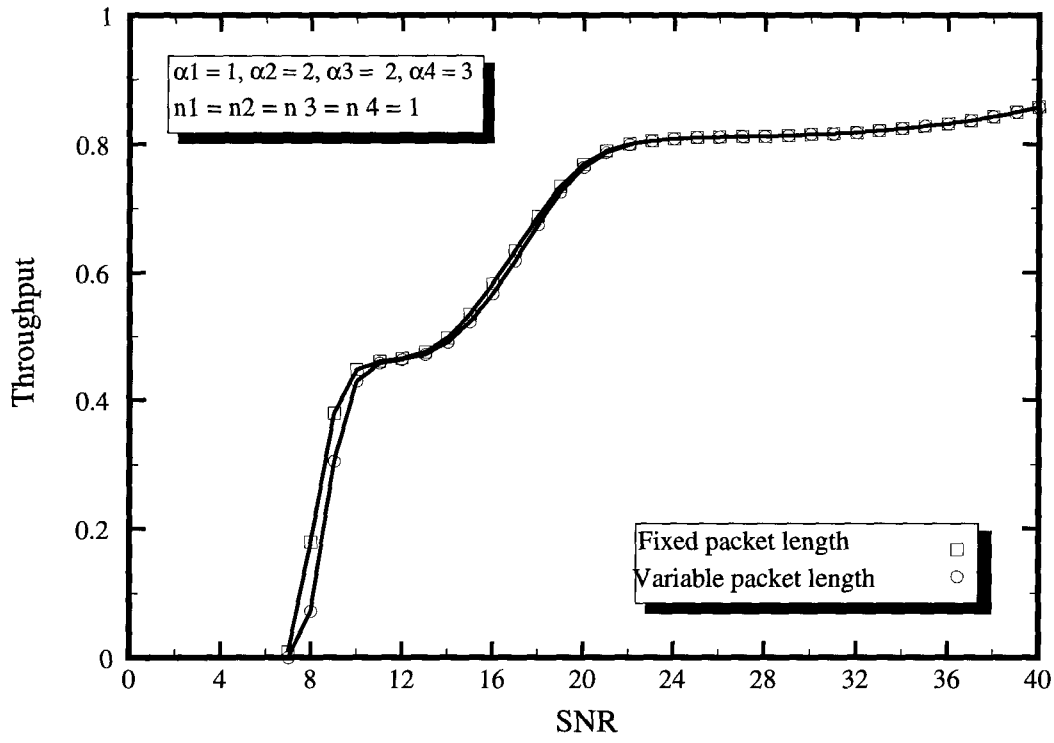

Figure 4.9 Throughput of the AHARQ with Fixed and Variable Packet Size

are shown in Figure 4.9, and it is seen the difference between the throughput performance of the scheme in the two cases is small. Although small packet size is desirable when re-transmissions are required, the constant number of error-detection bits included in the packet counteracts for this effect.

4.2 An adaptive Type-II ARQ Scheme with Code Combining Employing Complementary Convolutional Codes

The choice of optimal code rate for reliable data communication was investigated in the previous section. Even though the performance improvement was remarkable with the proposed adaptive scheme, the throughput dropped sharply for channels with extremely high

bit error rates. This undesirable outcome is caused by wasted retransmissions due to the disposal of erroneous packets at the receiver.

In this section , an adaptive Type-II HARQ with code combining employing the CPC codes is proposed and analyzed in the time-varying channels. Similar to the Adaptive Type-I HARQ, the protocol adopts an adaptive coding strategy which matches the code rate to the channel conditions. In addition, when multiple copies of the same packet encoded with equivalent codes of the same rate are received, they are combined to obtain a sequence with a low-rate code to combat the channel degradation. The system uses the same encoder/decoder devices but with each new retransmission a better code is resulted. Consequently, the scheme provides high throughput and reliability while keeping the system complexity at a minimum level.

4.2.1 System Operations

This is an extension to the Adaptive Type-I HARQ studied in the previous section. The system operates in the same manner except that the receiver takes advantage of the noisy packets to obtain significant data throughput. Like the adaptive Type-I scheme, the data to be transmitted is encoded with one of the M possible code rates corresponding to M channel states. The employed codes are high rate Punctured Convolutional Codes which are selected in such way that they satisfy the property of CPC codes [4].

In contrast to the previous adaptive scheme that repeats the same packet with the same rate and Perforation matrix while in state i, the transmitter of this system alternates between the equivalent codes of the same rate. The successive repetitions of the same packet are, then, stored and combined to obtain Complementary Codes which facilitate the recovery of

the message. The procedure for the transmission of equivalent codes is presented in detail in chapter 1.

To facilitate the process of combining equivalent codes, the encoded message is contained within a frame which consists of three parts. Each frame begins with a frame synchronization pattern of length F-bits, which identifies the start of a frame. Following that is the header which contains the routing information, the sequence and acknowledgment numbers, the information about data code rate and equivalent codes, and the length of the packet. The encoded message comprises the last part of the frame. The frame header is encoded separately form the message so that it may be read before making a decision as to whether to process the following data section. As mentioned in the preceding section on the Adaptive Type-I HARQ, the information about the code rate of the header is included after the frame synchronization pattern as a part of the header. The frame header code is the same as that used for the data except when retransmissions of the message is requested, the code rate of this field is changed to the desired one as determined by the channel state estimator. This measure ensures that the probability of header error rate is kept well below that of the message.

As for the message code rate, when the system determines a more powerful code is required to meet the prevailing channel condition, the message is sent with different equivalent codes of the same rate until it is successfully acknowledged. Then, the system switches to the suitable code rate. To incorporate these additional features in the channel state estimation algorithm and to make an efficient use of the code combining operation, we need to slightly modify the algorithm of the adaptive scheme described in the previous section. Therefore, the scheme with the inclusion of these factors can be summarized as follows.

1. Transmit at rate r_1. If there are n_1 frames in error in N transmitted frames, change the header code rate to r_2. However, continue transmitting the message with equivalent codes of the same rate r_1, until it is successfully received. Then, switch to rate r_2 for the next messages.
2. Transmit at rate r_i, $1 < i < M$, as long as the number of erroneous frames is less than n_i. Should n_i errors occur, switch to rate r_{i+1} for the frame header and continue transmitting the message with equivalent codes of the same rate r_i until an ACK is received. Then, transmit the next messages with rate r_{i+1}. In contrast, if there are less than n_i errors in N frames for α_i consecutive observation intervals, switch to rate r_{i-1} for the next frames.
3. Transmit at a code rate r_M as long as there are more than n_i errors in N frames. If retransmissions are required, send the frame header with rate r_M and continue transmitting the message with equivalent codes of the same rate r_M. If there are less than n_i errors in N frames for α_i consecutive observation intervals, then switch to rate r_{M-1} for encoding the next frames.

An analytical evaluation of the performance of this scheme is difficult since the probabilities of different error events during the code combining process are strongly dependant on each other. The comparison of this scheme to others is, therefore, done by computer simulations.

4.2.2 Simulation Results

The criteria for choosing the different system parameters are exactly the same as that for the adaptive Type-I HARQ. Therefore, the same adaptation interval and thresholds are chosen to ease the evaluation and investigation of the system performance. In this section,

the frame synchronization technique with threshold of 12, which is described in chapter 1, is utilized for detecting the beginning of a data frame.

To demonstrate the performance of the proposed algorithm, we consider the case in which the transmitter has four codes to choose from, each of which corresponds to a channel state. The codes are Punctured Convolutional codes (1, 1, 4), (5, 4, 4), (4, 3, 4), and (3, 2, 4) which represent the channel states 1, 2, 3, and 4 respectively. These codes are listed in Table 5, and they are examined to ensure that Complementary Codes can be obtained by combining process of equivalent codes. The information length is 900 bits, followed by 32 bits CRC for error detection. The header length is 57 bits including the CRC bits for error detection purposes. The values for the system parameters are selected according to the procedure outlined in the previous chapter:

$$\begin{aligned} N &= 8 \\ n_1 = 1, n_2 = 2, n_3 = 2, n_4 &= 1 \\ \alpha_1 = 1, \alpha_2 = 2, \alpha_3 = 2, \alpha_4 &= 3 \end{aligned} \tag{4.23}$$

Figure 4.10 shows the comparative results of the Adaptive Type-I and Type-II HARQ schemes. The same codes, adaptation thresholds, and observation intervals as explained above are also used to simulate the throughput curve of the Adaptive Type-I HARQ. Moreover, the frame structure is the same as that used in Adaptive Type-II HARQ schemes. It is observed that the proposed Type-II scheme provides a substantial throughput improvement over the Adaptive Type-I scheme at low SNR values. This improvement is attributed to combining noisy packets transmitted with equivalent codes.

Table 5 Distance Spectra of CPC codes (1, 1, 4), (5, 4, 4), (4, 3, 4), and (3, 2, 4)

R	**C_i**	**Perforation Matrix**	d_{free}	$(a_{d_{free}+j}, j = 0,1,2,...)$ $(c_{d_{free}+j}, j = 0,1,2,...)$
1	C_1	$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	—	— — — — — — — — — — — — — — — — — — — —
1	C_2	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$	—	— — — — — — — — — — — — — — — — — — — —
1/2	C_1+C_2	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	7	(6, 9, 12, 48, 111, 204, ...) (12, 36, 60, 216, 675, 1500, ...)
4/5	C_3	$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$	3	(3, 16, 103, 675, 3969, 24328, ...) (11, 78, 753, 6901, 51737, 386465, ...)
4/5	C_4	$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$	3	(3, 16, 103, 675, 3969, 24328, ...) (11, 78, 753, 6901, 51737, 386465, ...)
4/10	C_3+C_4	$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 \end{bmatrix}$	8	(2, 2, 5, 13, 19, 42, ...) (6, 4, 14, 53, 103, 236, ...)
3/4	C_5	$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/4	C_6	$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	3	(1, 2, 23, 124, 576, 2852, 14192, ...) (1, 7, 125, 936, 5915, 36608, ...)
3/8	C_5+C_6	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix}$	8	(1, 4, 3, 11, 18, 38, ...) (1, 11, 11, 45, 86, 204, ...)
2/3	C_7	$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$	4	(1, 0, 27, 0, 345, 0, ...) (1, 0, 124, 0, 2721, 0, ...)
2/3	C_8	$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$	4	(1, 0, 27, 0, 345, 0, ...) (1, 0, 124, 0, 2721, 0, ...)
2/6 1/3	C_7+C_8	$\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix}$	10	(3, 0, 18, 0, 36, 0, ...) (3, 0, 81, 0, 174, 0, ...)

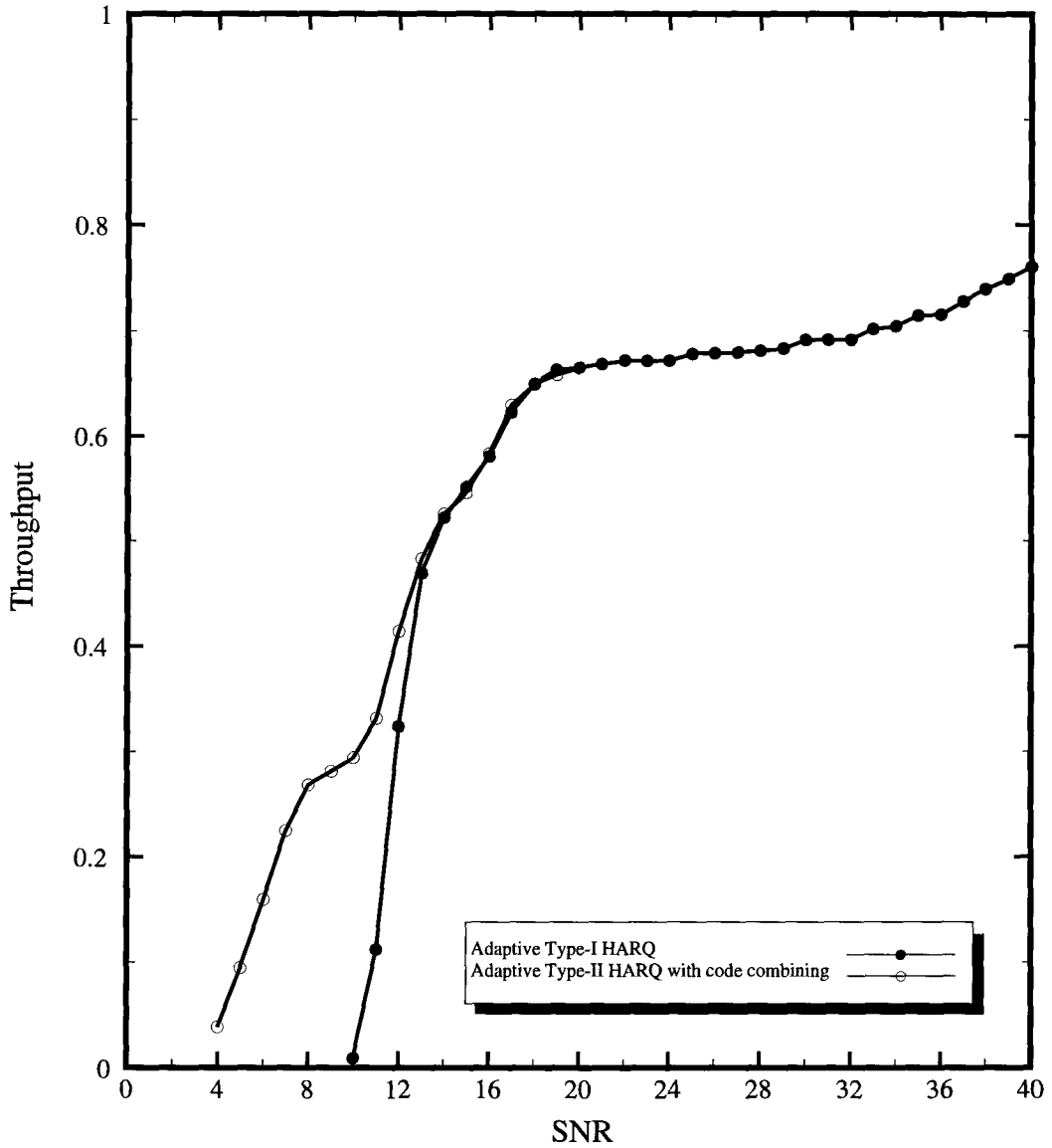

Figure 4.10 Throughput of the Proposed Adaptive Schemes

Computer Simulations of the proposed scheme in Figure 4.11 show that significant improvement is achieved over the Basic ARQ and Type-II HARQ schemes. The improvement in throughput increases up to 56% compared to the Type-II and Type-II HARQ with code combining for repetition codes [2].

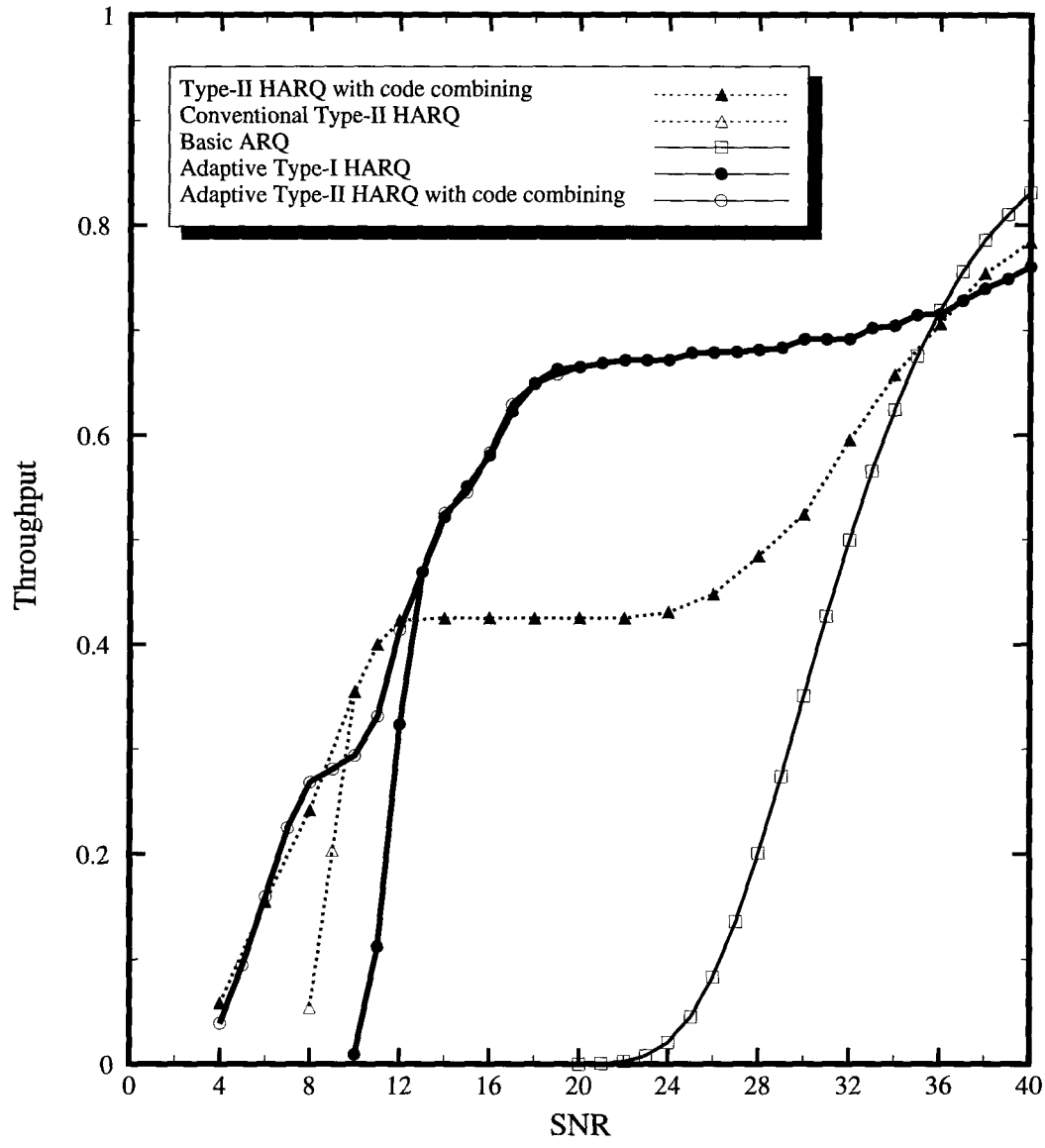

Figure 4.11 Throughput of Adaptive HARQ and Conventional ARQ Schemes

Chapter 5 Conclusions and Suggestions for Future Research

ARQ systems are typically analyzed with the assumption that perfect frame synchronization can be achieved. In other words, the fact that packets may be lost due to frame header failure is often ignored. Moreover, noiseless return channel is assumed to simplify the analysis of the schemes in question. We have studied and obtained the throughput expressions for the simple ARQ and MARQ schemes with the consideration of imperfect feedback channel and frame header sensitivity. It is found that these factors can result in the loss of packets and the occurrence of unnecessary retransmissions. Moreover, the advantage of ARQ with memory can be counteracted when frequent frame header loss brings about reduction in the amount of available repetition redundancy needed for further processing the packet. To enhance the system performance, it is recommended that the frame header and the ACK message be encoded or consecutively repeated a few times so that their error rate will be less than that of the data field after error correction or copy combining.

In the ARQ schemes with fixed-rate codes, the redundancy in the system may be more or less than the optimum value resulting in unsatisfactory performance at good or bad channel conditions. To remedy this problem, we have presented an adaptive scheme which is extremely effective for time-varying channels. An algorithm for estimating the channel conditions to facilitate code adaptation has been proposed. It has been illustrated that if the codes and system parameters are selected according to the actual channel states, significant throughput improvements can be achieved without affecting the system complexity. The

theoretical evaluation of the throughput of the adaptive Type-I HARQ has been conducted, and it has been shown that the theoretical results are consistent in terms of their agreement with simulations.

The drawback of the adaptive Type-I HARQ is that the packets detected in error are discarded even though they may contain useful information about the transmitted message. To take advantage of the additional retransmissions of a packet, we have proposed adaptive Type-II HARQ with CPC codes which yields further improvement compared to the adaptive Type-I HARQ. By means of simulations, we have demonstrated the resulting performance improvement due to combining equivalent codes. It is discovered that the Type-II HARQ always performs remarkably well over the whole range of SNR values compared to all other ARQ schemes.

For future research, we suggest the following:

- Implement and investigate the effects of header and ACK sensitivities on the system performance and compare the results to those obtained from the theoretical and simulation cases.
- Study the practical implementation of the adaptive Type-I and Type II HARQ schemes.
- Study more effective methods of monitoring the channel conditions to minimize the channel estimation delays in the transmitter.

References

[1] S. Lin and J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] S. Kallel, "Analysis of a Type-II Hybrid ARQ Scheme With Code Combining," *IEEE Transactions on Communications*, pp. 1133–1137, Aug 1990.

[3] D. Chase, "Code Combining. A Maximum-Likelihood Decoding Approach for Combining an Arbitrary Number of Noisy Packets," *IEEE Trans. Commun.*, vol. COM-33, pp. 385–393, May 1985.

[4] S. Kallel, "Complementary Punctured Convolutional Codes and Their Use in Hybrid ARQ Schemes," *IEEE Pacific Rim Conference*, pp. 186–189, May 1993.

[5] S. Kallel and C. Leung, "Efficient ARQ Schemes with Multiple Copy Decoding," *IEEE Transactions on Communications*, pp. 642–650, March 1992.

[6] S. Kallel, "Analysis of Memory and Incremental Redundancy ARQ Schemes Over a Nonstationary Channel," *IEEE Transactions on Communications*, pp. 1474–1480, September 1992.

[7] G. Benelli, "A Go-Back-N Protocol for Mobile Communications," *IEEE Transactions on Vehicular Technology*, pp. 715–720, March 1991.

[8] N. Shacham and A. Livne, "An Adaptive Hybrid ARQ Algorithm for Radio Channels," *IEEE International Communications Conference*, vol. 3, pp. 1390–1394, 1985.

[9] P. Elias, "Coding for Noisy Channels," *IRE Conv.Rec.*, pp. 37–47, 1955.

[10]A. J. Viterbi, "Error Bounds for Convolutional Codes and Asymptotically Optimum Decoding Algorithm," *IEEE Trans. Info. Theory*, vol. IT13, pp. 260–269, April 1967.

[11]R. Ziemer and W. Tranter, *Principles of Communications*. Houghton Mifflin, 1985.

[12]J. Cain, G. Clark, and J. Geist, "Punctrured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding," *IEEE Transactions on Information Theory.*, vol. IT-25, pp. 97–100, Jan 1979.

[13]Y. Yasuda, K. Kashiki, and Y. Hirata, "High Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding," *IEEE Transactions on Communications*, pp. 315–319, March 1984.

[14]S. Kallel and D. Haccoun, "Generalized Type-II Hybrid ARQ Scheme Using Punctured Convolutional Coding," *IEEE Transactions on Communications*, pp. 1938–1946, November 1990.

[15]J. Hagenauer, "Rate-Compatible Punctured Codes (RCPC Codes) and their Applications," *IEEE Transactions on Communications*, pp. 389–399, April 1988.

[16]S. Kallel and C. Leung, "An Adaptive Incremental Redundancy Selective Repeat ARQ Scheme for Finite Buffer Receivers," *Proc. INFOCOM 1991, Miami, FL*, pp. 720–725, May 1991.

[17]W. Jakes, *Mobile Communications Engineering*. McGraw-Hill, 1982.

[18]J. Proakis, *Digital Communications*. McGraw-Hill, New York, 1983.

[19]A. Sastry, "Performance of Hybrid Error Control Schemes on Satellite Channels," *IEEE Trans. on Communications*, vol. COM-23, pp. 689–694, July 1975.

[20]H. Yamamto and K. Itoh, "Viterbi Decoding Algorithm for Covolutional Codes with Repeat Requests," *IEEE Trans. Inform. Theory.*, vol. IT-26, pp. 540–547, Sep 1980.

[21]A. Drukarev and C. D, "Hybrid ARQ Error Control using Sequential Decoding," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 521–535, July 1983.

[22]C. Leung and A. Lam, "Forward Error Correction for an ARQ Scheme," *IEEE Trans on Communications*, vol. COM-29, pp. 1514–1519, Oct 1981.

[23]D. Mandelbaum, "Adaptive-Feedback Coding Scheme using Incremental Redundancy," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 388–389, May 1974.

[24]J. Metzner, "Improvements in Block-Retransmission Schemes," *IEEE Trans on Communications*, vol. COM-27, pp. 525–532, Feb 1979.

[25]P. Driessen, "Binary Frame Synchronization Sequences for Packet Radio," *Electronics Letters*, vol. 23, pp. 1190–1191, Oct 1987.

[26]H. Kwon, "Frame Synchronization for a Channel with Different Data Rates," *IEEE MILCOM*, vol. V.1, pp. 176–180, 1990.

[27]R. Lugand, D. Costello, and R. Deng, "Parity Retransmission Hybrid ARQ Using Rate 1/2 Convolutional Codes on a Nonstationary channel," *IEEE Transactions on Communications*, pp. 755–763, July 1989.

[28]D. Haccoun and G. Begin, "High-Rate Punctured Convolutional Codes for Viterbi and Sequential Decoding ," *IEEE Transactions on Communications*, vol. 37, pp. 1113–1124, November 1989.

[29]T. Sato, M. Kawbe, T. Kato, and A. Fukasawa, "Protocol Configuration and Verification

of an Adaptive Error Control Scheme Over Analog Cellular Networks," *IEEE Transactions on Vehicular Technology*, vol. 41, pp. 69–76, February 1992.

[30]M. Kousa and M. Rahman, "An Adaptive Error Control System Using Hybrid ARQ Schemes," *IEEE Transactions on Communications*, vol. 39, pp. 1049–1057, July 1991.

[31]M. Ross, *Stochastic Processes*. John, Wiley, and sons, Inc. London, 1983.