Mobile Networks and Applications 2 (1997) 167–182

Error control schemes for networks: An overview

Hang Liu [a], Hairuo Ma [a], Magda El Zarki [a] and Sanjay Gupta [b]

[a] *Department of Electrical Engineering, University of Pennsylvania, 200 S. 33rd Street, Philadelphia, PA 19104, USA*
[b] *Motorola, GSM Products Division, 1501 West Shure Drive (IL27-3205), Arlington Heights, IL 60004, USA*

In this paper, we investigate the issue of error control in wireless communication networks. We review the alternative error control schemes available for providing reliable end-to-end communication in wireless environments. Through case studies, the performance and tradeoffs of these schemes are shown. Based on the application environments and QoS requirements, the design issues of error control are discussed to achieve the best solution.

1. Introduction

In recent years there has been an increasing trend towards personal computers and workstations becoming portable. Desire to maintain connectivity of these portable computers to the existing installation of Local Area Networks (LANs), Metropolitan Area Networks (MANs), and Wide Area Networks (WANs), in a manner analogous to present day computers, is fueling an already growing interest in wireless networks. Wireless networks will be needed to provide voice, video and data communication capability between mobile terminals and also to permit such terminals to have access to wireline networks. However, before wireless networks can be employed for packet voice, video, data, and other applications, it is important that appropriate communication protocols suited for the wireless environment are developed. Of specific interest are "physical", "link" and "network" layer protocols that take into account the characteristics of the underlying communication channel.

Wireless channels provide error rates that are typically around 10^{-2}. Such high error rates result due to multipath fading which characterize mobile radio channels. However, many applications such as video and data transmissions require that the error rates be significantly smaller. In addition to the poor channel quality, the design of wireless communication systems is complicated by the rapidly changing quality of the radio channel.

To increase the apparent quality of a communication channel there exist the following two distinct approaches:

- *Forward Error Correction* (FEC) which employs error correcting codes to combat bit errors (due to channel imperfections) by adding redundancy (henceforth *parity bits*) to information packets before they are transmitted. This redundancy is used by the receiver to *detect* and *correct* errors.
- *Automatic Repeat Request* (ARQ) wherein only error detection capability is provided and no attempt to correct any packets received in error is made; instead it is requested that the packets received in error be retransmitted.

FEC and ARQ are two basic categories of error control techniques. ARQ is simple and achieves reasonable throughput levels if the error rates are not very large. However, in its simplest form, ARQ leads to variable delays which are not acceptable for real-time services. FEC schemes maintain constant throughput and have bounded time delay. However, the post decoding error rate rapidly increases with increasing channel error rate. In order to obtain high system reliability, a variety of error patterns must be corrected. This means that a powerful long code is necessary, which makes the coder-decoder pair hard to implement and also imposes a high transmission overhead. Further complicating matters is that the wireless channel is non-stationary, and the channel bit error rate varies over time. Typical FEC schemes are stationary and must be implemented to guarantee a certain Quality Of Service (QOS) requirement for the worst case channel characteristics. As a consequence, FEC techniques are associated with unnecessary overhead that reduces throughput when the channel is relatively error free.

In order to overcome their individual drawbacks, the combination of these two basic classes of error control schemes, called hybrid ARQ schemes, have been developed [12,13,31,32].

Specifically, this paper examines the alternatives available for providing a reliable end-to-end communication channel in wireless networks and discusses their impact on network design and protocols. The paper is organized as follows: In sections 2 and 3, the two fundamental techniques for error detection and correction, FEC and ARQ are reviewed, respectively. In section 4 we discuss hybrid error control techniques, i.e., schemes that combine FEC and ARQ in their efforts to construct a reliable pipe for data transfer. In section 5, we present case studies and examples from existing and future networks that highlight the merits and demerits of the above error control methods and discuss the relevance of each for various practical applications. Finally, the discussion is summarized in section 6.

2. Forward error correction

Forward error correction involves addition of redundant bits (henceforth referred to as *parity* bits), that are used to aid in correcting any bits that are received in error. Shannon's channel coding theorem [8] states that there always exists a coding scheme that enables information to be transmitted over any given channel with arbitrarily small error probabilities provided the data rate (including that due to parity bits) over the channel is less than the channel capacity (as defined by the classical Shannon's theorem). Over the last four decades a number of powerful and efficient codes have been designed [8,16,19,23].

2.1. Block coding

Block coding schemes divide a bit stream into non-overlapping blocks and code each block independently. Block codes used in practical applications today belong to the class of linear cyclic codes, since these codes lend themselves to easier implementations. A coding scheme is referred to as being linear if the sum of two code vectors is also a code vector. Similarly, a coding scheme is referred to as being cyclic if all cyclic shifts of a code vector results in a valid code vector. Binary Bose–Chaudhuri–Hocquenghem (BCH) codes and non-binary Reed–Solomon (RS) codes are two kinds of widely used linear cyclic block codes.

2.1.1. Bose–Chaudhuri–Hocquenghem (BCH) codes

For any positive integers, $m \geqslant 3$ and $t < 2^{m-1}$, there is a binary BCH code with the following parameters, (referred to as an (n, k, t) BCH code):

- block length: $n = 2^m - 1$,
- number of parity check bits: $n - k \leqslant mt$,
- minimum distance: $d_{\min} \geqslant 2t + 1$.

Each binary BCH code (n, k, t) can correct up to t-bit errors, and thus it is also referred to as a t-error-correcting code.

2.1.2. Reed–Solomon (RS) codes

The binary BCH codes can be generalized to non-binary codes. If p is a prime number and q is any power of p, there exist BCH codes with q-ary symbols. For any choice of positive integer s and t, a q-ary BCH code is of length $n = q^s - 1$, which is capable of correcting any combination of t or fewer symbol errors and requires no more than $2st$ parity-check symbols. RS codes are a subclass of non-binary BCH codes with $s = 1$. A (n, k, t) RS code with q-ary symbols has the following parameters:

- block length: $n = q - 1$,
- number of parity-check bits: $n - k = 2t$,
- minimum distance: $d_{\min} = 2t + 1$.

An (n, k, t) RS code is capable of correcting any combination of t or fewer symbol errors. In practical applications, RS codes with code symbols from $q = 2^m$ are chosen.

BCH and RS block coding schemes have a well defined algebraic structure, which has facilitated the development of efficient coding and decoding schemes. In addition, RS codes have optimal "distance properties", i.e., provide optimal error correction capability given a fixed number of parity bits, and excellent "burst error suppression" capabilities.

2.2. Code shortening

Often, a block code of desirable natural length or suitable number of information digits may not exist. In this case, *code shortening* is performed, which involves choosing a code with block length greater than the required length and subsequently shortening it to meet the requirement. Code shortening is most easily done by setting a selected number of the information symbols to zero in the encoding operation. For example, given an (n, k) code C, consider the set of code vectors for which the b leading high-order information symbols are equal to zero. Such code vectors form a subset of code C. If the b zero information symbols are deleted from each of these code vectors, we obtain a set of vectors of length $n - b$. These shortened vectors form an $(n-b, k-b)$ code. The error detection and correction capability of the shortened code is at least as great as the code from which it was derived. For RS codes, the minimum distance is unchanged after shortening.

2.3. Convolutional codes

Block coding schemes are frequently referred to as memoryless since successive information blocks are coded independently. *Convolutional codes* are a popular class of coders with memory, i.e., the coding of an information block is a function of the previous blocks. An (n, k, m) convolutional code generates n encoded bits for every k information bits, where m refers to the memory of the encoder. The n encoder outputs at any given time instant, depend not only on the k inputs but also on m previous input blocks (where $(m + 1)$ is sometimes referred to as the code constraint length) [16]. Figure 1 shows a rate $1/2$ binary convolutional encoder with $m = 4$. For each input bit, there are 2 output bits which depend on the previous 4 input bits. The encoder consists of an m-stage shift register together with n modulo-2 adders and a multiplexer for serializing the encoder outputs. If the input sequence is $\mathbf{u} = (u_0, u_1, u_2, \ldots)$, the two encoder output sequences $\mathbf{v}^{(1)} = (v_0^{(1)}, v_1^{(1)}, v_2^{(1)}, \ldots)$ and $\mathbf{v}^{(2)} = (v_0^{(2)}, v_1^{(2)}, v_2^{(2)}, \ldots)$ are equal to the convolution of the input sequence $\mathbf{u}$ with the two code generator sequences $\mathbf{g}^{(1)} = (1, 0, 0, 1, 1)$ and $\mathbf{g}^{(2)} = (1, 1, 1, 0, 1)$, i.e., the encoding equations are $\mathbf{v}^{(1)} = \mathbf{u} * \mathbf{g}^{(1)}$ and $\mathbf{v}^{(2)} = \mathbf{u} * \mathbf{g}^{(2)}$.

In general, for convolutional codes, each information sequence (input, output or code generator sequence) can be written as a polynomial with the coefficients of the polynomial being equal to the elements of the corresponding sequence. In this case the output polynomial is equal to

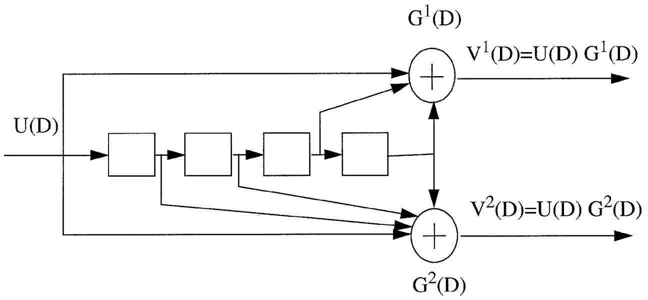


Figure 1. Rate $1/2$ convolutional encoder with $m = 4$.

the product of the input polynomial and generator polynomial. For the $(2, 1, 4)$ convolutional code discussed above, the encoding equations are

$$V^{(i)}(D) = U(D)G^{(i)}(D), \quad i = 1, 2, \tag{1}$$

where

$$\begin{aligned} U(D) &= u_0 + u_1 D + u_2 D^2 + \cdots, \\ V^{(1)}(D) &= v_0^{(1)} + v_1^{(1)} D + v_2^{(1)} D^2 + \cdots, \\ V^{(2)}(D) &= v_0^{(2)} + v_1^{(2)} D + v_2^{(2)} D^2 + \cdots, \end{aligned}$$

and

$$\begin{aligned} G^{(1)}(D) &= 1 + D + D^4, \\ G^{(2)}(D) &= 1 + D^2 + D^3 + D^4 \end{aligned}$$

are referred to as the generator polynomials of the code. After multiplexing, the code vector is

$$V(D) = V^{(1)}\left(D^2\right) + DV^{(2)}\left(D^2\right).$$

Encoding for both linear cyclic block codes and linear convolutional codes involves simple arithmetic operations and therefore they are easily implemented. If a block code is used for error detection, only simple integer division is needed; however, decoding block codes or convolutional codes for error correction is much more tedious. For block codes, an iterative algorithm is often used to correct the errors. Error correction algorithms become quite complex for long codes with large error correction capability; especially, for non-binary codes. For convolutional codes, decoders are often based on the Viterbi algorithm which is known to be an optimal algorithm [16]. Its decoding complexity grows exponentially with code memory length. Therefore it is effective for short memory length codes. Many other decoding algorithms such sequential decoding, majority-logic decoding, etc. have also been developed; interested readers are referred to [8,19] for a more detailed discussion.

2.4. Code puncturing

The characteristics of a wireless channel typically vary with time, and therefore to obtain optimal performance it is necessary to adapt the error coding scheme to the changing channel characteristics. *Code puncturing* allows an encoder/decoder pair to change code rates, i.e., code error correction capabilities, without changing their basic structure [11–13]. Code puncturing involves not transmitting (i.e., deleting) certain code bits. It is important to note that both convolutional codes and block codes can be punctured.

Punctured convolutional codes were first introduced by Clark et al. [9]. Hagenauer modified the concept of punctured convolutional codes for the generation of a family of rate compatible punctured convolutional (RCPC) codes by adding a rate-compatibility restriction to the puncturing rule [12]. The rate-compatibility restriction implies that all the code bits of a high rate code of the family are used by the lower rate codes. These codes are attracting more and more attention because of their flexibility.

We now discuss in some detail the process of puncturing codes. A low rate $1/n$ convolutional code (called mother code) is periodically punctured with period p to obtain a family of codes with rate p/v, where v can be varied between $p + 1$ and np. As an example, we consider punctured convolutional codes obtained from a rate $1/3$ mother code. To generate a rate p/v punctured convolutional code ($p/v > 1/3$), we delete $(3p - v)$ bits from every $3p$ code bits corresponding to the encoded output of p information bits by the original rate $1/3$ code. The resulting rate is then equal to the desired rate $r = p/v$. The punctured codes have the same number of states as the mother code, i.e., the same memory length m. The deleted bit pattern must be carefully chosen to obtain desirable performance.

We now illustrate code puncturing with the help of an example. Consider the rate $1/3$ mother code defined by the generator polynomials: $G^{(1)} = D^4 + D + 1$, $G^{(2)} = D^4 + D^3 + D^2 + 1$ and $G^{(3)} = D^4 + D^2 + D + 1$. Its puncturing matrices with a puncturing period of 8 are given in table 1 [12]. Each matrix has 8 columns and 3 rows, corresponding to the puncturing cycle and the branches at the output of the rate $1/3$ coder, respectively. The elements of puncturing matrices are only zeros and ones. A zero in a puncturing matrix means that the corresponding code bit will not be transmitted, a one means that it is inserted in the channel bit stream. For example, to generate a $8/22$ code, puncturing matrix $p(8/22)$ is used. Encoding of 8 information bits with the three generator polynomials results in 3×8 intermediate bits at the three output branches. Every fourth and eighth output bit of the third branch is deleted. Instead of transmitting 3×8 bits, only 22 bits are transmitted per 8 information bits. Therefore, a rate 8/22 code is generated. In general, the puncturing matrix $p(r = p/v) = [p_{ij}]$ for a mother code of rate $1/n$ and a puncturing period of p has n rows and p columns. The number of zeros in the puncturing table is equal to $np - v$.

Two punctured convolutional codes, obtained from the same mother code, are said to be rate-compatible if all the code bits in the higher rate code are used in the lower rate codes. Let $p(r_1)$ and $p(r_2)$ be the puncturing matrices of two rate-compatible codes ($p/p{+}1 > r_1 > r_2$). If an element in $p(r_1)$ is equal to one ($p_{ij}(r_1) = 1$), then the same element in $p(r_2)$ is also equal to one ($p_{ij}(r_2) = 1$). Given a high-rate punctured convolutional code with puncturing matrix $p(r_1)$,

Table 1
Puncturing table for RCPC codes with mother code rate $1/3$, memory $m = 4$ and period $p = 8$.

Code rate r	8/9	8/10	8/12	8/14	8/16	8/18	8/20	8/22	8/24
Puncturing matrix	1111 0111 1000 1000 0000 0000	1111 1111 1000 1000 0000 0000	1111 1111 1010 1010 0000 0000	1111 1111 1110 1110 0000 0000	1111 1111 1111 1111 0000 0000	1111 1111 1111 1111 1000 1000	1111 1111 1111 1111 1100 1100	1111 1111 1111 1111 1110 1110	1111 1111 1111 1111 1111 1111

a lower rate-compatible punctured convolutional code can be generated if we replace some zeros of $p(r_1)$ with ones and retain the previous ones in $p(r_1)$. For example, if we replace the fifth zero of the first row in the puncturing matrix of a rate $8/9$ code ($p(8/9)$ given in table 1) with a one, the puncturing matrix of a rate 8/10 compatible code will be obtained.

The encoder for a punctured code can be fabricated using the original low-rate convolutional encoder followed by a bit selector which deletes specific code bits according to a given puncturing rule. Only the bit selection rule is changed to generate different rates of codes. At the receiver side, a Viterbi decoder based on the mother code decoder is used for decoding the punctured codes of the family. To decode different rate codes, only metrics are changed according to the same puncturing rule used by the encoder (the deleted bits are not counted when calculating the path metrics). RCPC codes have been shown to offer nearly equivalent performance when compared with the best previously known codes of the same rate. However, they are much simpler to decode than conventional convolutional codes.

2.5. Code selection

Having discussed various coding schemes, we now consider criteria that must be taken into account when selecting a FEC scheme for any given application.

1. *Probability of uncorrected errors*: Since it is impossible for any coding scheme to detect all errors and correct them, it is important to choose coding schemes for which the probability of both undetectable and uncorrectable (but detectable) errors is minimized (or satisfies the application under consideration).
2. *Overhead*: The FEC codes should add as little as possible overhead and maximize the code rate. However, increased code capability generally leads to lower code rate.
3. *Complexity*: The implementation complexity of the coding/decoding scheme which typically increases with increase in code length and its capability to detect and correct errors.

Before proceeding further we present some numerical results and illustrate the impact FEC has on various performance characteristics. Let's first consider the impact of using binary BCH codes on a channel that is characterized by random errors.

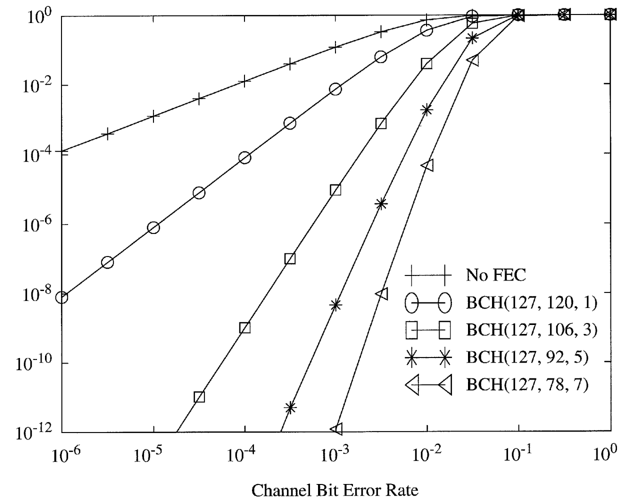


Figure 2. BER performance of BCH codes over random error channels.

Figure 2 shows the decoding performance versus average channel SNR for BCH(127, 120, 1), BCH(127, 106, 3), BCH(127, 92, 5), and BCH(127, 78, 7) codes. The associated overheads are 5.5%, 16.5%, 27.5%, and 38.5%, respectively. As expected, increased code error correction capability reduces decoding error rate but requires more redundancy.

For any given code rate (ratio of parity bits and block size) the error correction capability of a block coding scheme increases with increase in block size. Therefore, a larger block size enables one to utilize the channel more efficiently. Larger block codes also help mitigate the effect of burst errors, that often characterize wireless channels. However, the use of larger block sizes creates several problems. First, the *packetization delay*[1] increases with increase in block size, and this limits the maximum block size that can be used in various applications. Second, the implementation complexity of a coding system grows exponentially with increase in the block length for block codes or with memory length for convolutional codes, limiting codeword lengths.

Considerable effort has been directed towards implementing long, powerful codes with minimal complexity. Cascading of two or more codes was proposed as a means of constructing long codes which could be encoded and decoded in a simple manner [8,14,16,19]. A simple *concatenated code* is formed from two codes in series. The code closer to the channel is called the inner code, whereas the code outside the inner code is known as the outer code. Fig-

[1] The time required to collect data bits to form a block of desired size.

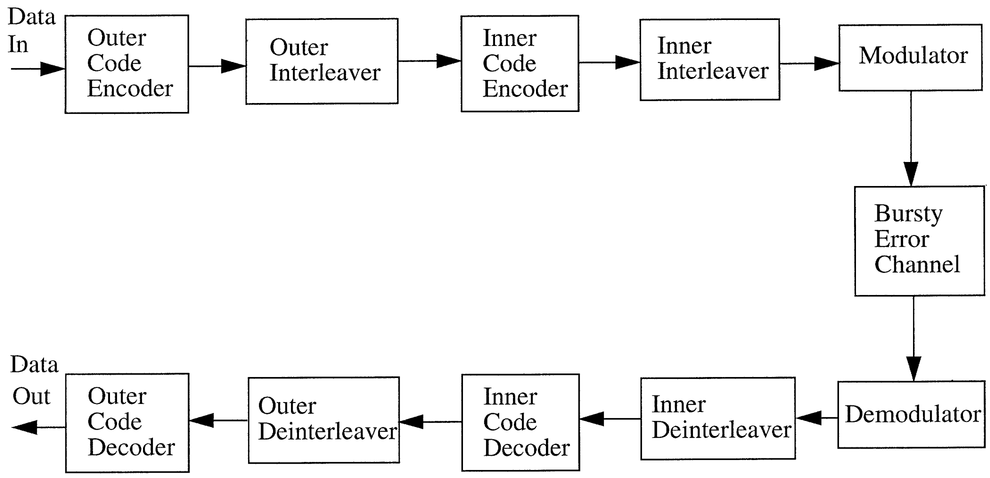


Figure 3. Block diagram of communication system using a concatenated code.

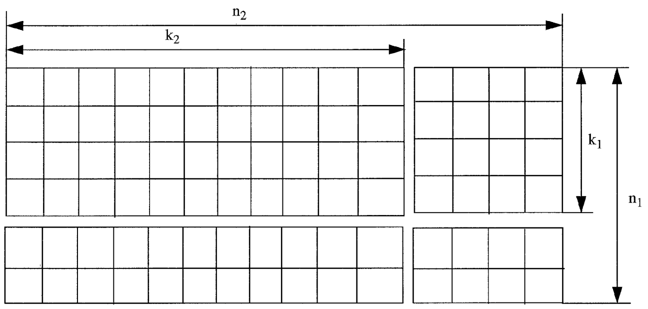


Figure 4. Structure of product codes [28, figure 4.71, p. 479].

ure 3 shows the diagram of a concatenated coding system that employs two interleavers. (Interleavers are described in detail in the next subsection.) The inner interleaver randomizes bursty channel errors. The outer interleaver is used because the errors resulting from the inner decoder are bursty in nature. The other method to form a long code while still maintaining coder-decoder simplicity is to use a product code [16,28]. Typically, product codes are obtained by encoding the information symbols in two dimensions; k_1k_2 information symbols are arranged in a matrix with k_1 rows and k_2 columns. Each row is encoded by an (n_2, k_2, t_2) code and each column is encoded by an (n_1, k_1, t_1) code. Then the information matrix with the dimensions $(k_1 \times k_2)$ is encoded to the dimension of $(n_1 \times n_2)$. Figure 4 shows an example of a product code by using RS(15, 11, 2) code for the rows and the shortened RS(6, 4, 1) for the columns where the symbol size is 4 bits for both the codes [28].

The performance of a specific code depends on the coding rate, the code length, the modulation schemes, the channel error patterns, etc. It is difficult to choose the best code for all the different channel conditions and system requirements. Figure 5 gives a general comparison for several different coding schemes over Rayleigh fading channels with MSK modulation:

(1) RS(57, 29, 14) code with a symbol size of 8 bits and no interleaving (the block size is 456 bits),

(2) RS(12, 6, 3) with a symbol size of 4 bits and block symbol interleaving over 9 codewords (the block size is 432 bits),

(3) (2, 1, 4) convolutional codes with bit interleaving (the block size is 448 bits) [28].

All the coding schemes here have a code rate about $1/2$. The BER performance curves of the two RS codes have a cross-over at a channel SNR of 21 dB. Below 21 dB, the shorter RS(12, 6, 3) code is better than the longer RS(57, 29, 14) code. The longer code having a larger symbol field, suffers a higher channel symbol error rate that is likely to cause more decoding errors. This results in a worse performance. Above 21 dB, the longer RS(57, 29, 14) code yields a better performance than the shorter RS(12, 6, 3) code because it has more parity symbols in a codeword and better burst-dispersing properties, that is, a higher correcting capability, The convolutional code results in a BER performance better than both RS codes. This can be contributed to using soft-decision Viterbi decoding which makes use of the channel information.

Figure 6 compares the performance of concatenated and product codes [28]. The product code uses a RS(15, 11, 2) code and a RS(6, 4, 1) code with a symbol size of 4 bits. Two concatenated coding schemes are considered:

(1) RS(48, 36, 6) with an 8-bit symbol as the outer code and a rate $2/3$ convolutional code obtained by puncturing a (3, 1, 4) mother code as the inner code,

(2) RS(48, 36, 6) with an 8-bit symbol as the outer code and a rate $2/3$ convolutional code obtained by pucturing a (3, 1, 6) mother code as the inner code.

The performance of the product code is poor because the codes used for both dimensions are very weak (double error correcting code for the rows and single error correcting codes for the columns). About 0.5 dB gain can be obtained when the mother code memory length increases from 4 to 6 for the two concatenated coding schemes.

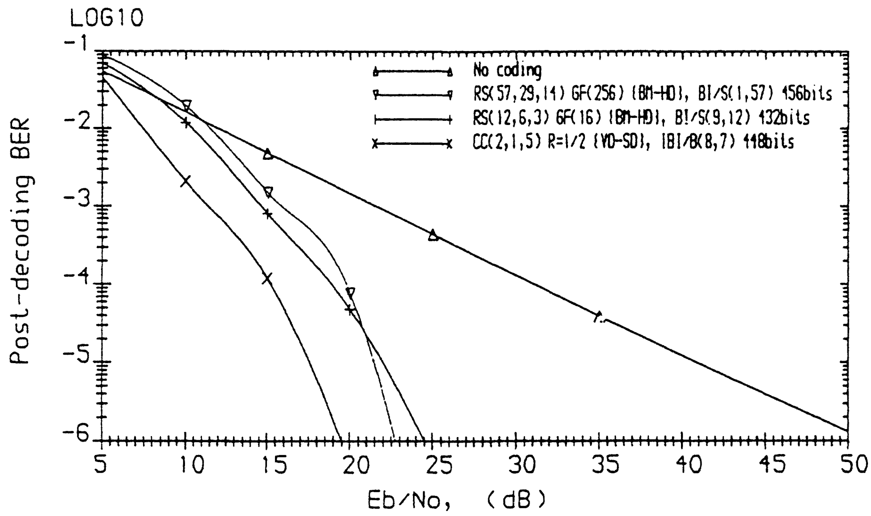


Figure 5. BER performance of different codes over Rayleigh fading channel [28, figure 4.72, p. 481].

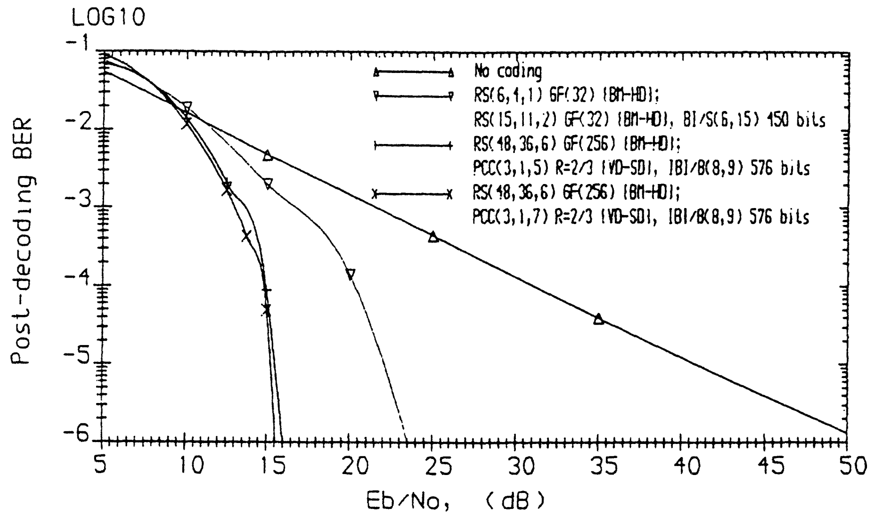


Figure 6. BER performance of concatenated and product codes over Rayleigh fading channel [28, figure 4.73, p. 481].

2.6. Interleaving

An alternative to choosing long codes to combat the effect of burst errors is interleaving. Interleaving simply involves interleaving symbols from two or more codewords before transmission on the channel. The number of codewords that are interleaved is referred to as the *depth* of the interleaver, m. Figure 7 shows a interleaver with a interleaving depth of m and a codeword length of N. The data is written row-by-row into a $m \times N$ matrix and read out column-by-column by the interleaver before sending it over the channel. The reverse process is performed at the deinterleaver. Therefore, between successive symbols of any given codeword there are $m - 1$ symbols that belong to the $m - 1$ other codewords being interleaved. If the interleaver has sufficient depth the fading processes that affect successive symbols belonging to the same codeword will be uncorrelated. Therefore, from the perspective of any single codeword, interleaving makes a burst error channel appear as one which has only random errors. Note that interleaving does not decrease the long-term bit error rate but it is successful in decreasing the number of errors in each codeword, therefore the codeword should have enough capability to correct the erroneous symbols in it after deinterleaving.

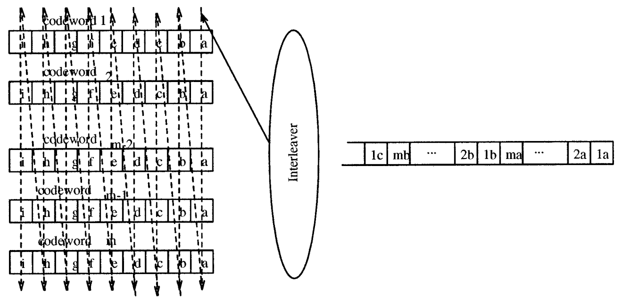


Figure 7. Interleaver.

Results in [34] show that the FEC and interleaving strategy is effective when tm exceeds $1/r$ where t is the code error correction capability and $1/r$ is the average burst length.

Note that interleaving results in extra delay because deinterleaving can be started only after all the interleaved data is received. For the above example, a delay on the order of $1/rt$ codewords is introduced.

3. Automatic Repeat Request (ARQ)

ARQ is an error control mechanism that relies on retransmitting data that is received with errors [20]. In such schemes, messages are divided into blocks of suitable size, that are transmitted after a small number of parity bits have been added. At the receiver these parity bits are used to detect the presence of errors in the received packet. In case errors are detected in a received packet, the receiver requests a retransmission of the packet.

Automatic Repeat Request (ARQ) protocols roughly operate as follows: The transmitter numbers the packets to be transmitted sequentially (using numbers from a finite set) and maintains a timer for each packet it transmits. The receiver acknowledges, at the very least, the receipt of each successful packet (a packet that is received with no errors) by transmitting a packet, referred to as an ACK bearing the sequence number of the packet being acknowledged. Packets that have not been successfully acknowledged, i.e., an ACK has not been received, in a predetermined time interval, henceforth referred to as *timeout*, are assumed to be lost (or corrupted) and are retransmitted.

In some cases, negative acknowledgements (NAKs) are transmitted by the receiver for every packet received in error. If NAKs are employed, a packet is retransmitted following the receipt of a negative acknowledgement. Since some of the transmitted packets can be lost or misrouted, NAKs can not be transmitted for these lost packets. We now briefly describe three of the most popular ARQ protocols – Stop and Wait, Selective Repeat, and Go-Back-N.

3.1. Stop and Wait

When using the *Stop and Wait* (SW) ARQ protocol, the DLC protocol transmits a packet only when all previously transmitted packets have been successfully acknowledged. Hence, when using SW, the transmitter after transmitting a packet waits for its acknowledgement. Once its acknowledgement has been received the next packet is transmitted. However, if an acknowledgement does not arrive until a timeout timer expires, the packet is retransmitted. Therefore, in SW there is never more than a single packet that is unacknowledged at any given instant of time. Since the transmitter does not use the available channel during time intervals it waits for an ACK, the maximum data transfer rate that can be supported is limited. This limits cases where the SW ARQ protocol can be employed.

3.2. Selective Repeat

Unlike SW, when using *Selective Repeat* (SR), packets, if available, are transmitted continuously by the DLC layer. As before, the receiver acknowledges each successfully received packet by transmitting an ACK bearing the sequence number of the packet being acknowledged. If an acknowledgement is not received for a packet before the expiration of the timeout, the packet is retransmitted. Once a packet has been retransmitted the transmitter resumes transmission of packets from where it left off, i.e., if a is the packet with the largest sequence number that has been transmitted, packet with sequence number $a + 1$ is transmitted next (assuming that no other timers have expired in the meantime). Since when the SR ARQ protocol is employed, packets are continuously being transmitted the inefficiency associated with SW is eliminated. Observe that when SR is employed packets can be accepted out of sequence. Hence, packets received out of sequence have to be buffered and sequenced before they can be delivered.

3.3. Go-Back-N

When *Go-Back-*N (GBN) is employed, packets are transmitted continuously as in SR. However, at the receiver, the DLC layer accepts packets only in the order in which they were transmitted. Packets received out of sequence are discarded and not acknowledged. Since the receiver accepts packets only in-sequence, after a timeout, the transmitter retransmits the packet that timed out and all packets with sequence numbers that follow the one that was retransmitted. Hence, each time a timeout occurs all packets that are yet to be acknowledged are retransmitted. It is important to observe that GBN attempts to combine the desirable features of SR and SW, i.e., packets are transmitted continuously, as in SR, without the need to buffer out of sequence packets and there is no resequencing overhead.

4. Hybrid error control

Given that neither FEC nor ARQ alone can deliver the desired performance it is necessary that *hybrid ARQ error control* schemes (that use both FEC and ARQ) be used. There are two types of hybrid ARQ schemes, type-I and type-II. *Type-I hybrid* ARQ includes parity bits for both error detection and error correction in every transmitted packet, using either a single code designed for simultaneous error correction and detection or a correction code as the inner code along with a detection code as the outer code. If the number of erroneous symbols in a received codeword is within the error correction capability of the code, the errors are corrected. If an uncorrectable error pattern is detected, the receiver discards the received codeword and requests a retransmission. The transmitter retransmits the same codeword. When the retransmitted codeword is received, the decoder again attempts to correct the errors within the error-correction capability of the code. If the packet arrives with detectable but uncorrectable errors a retransmission is requested again. This process continues until the codeword is successfully accepted or the maximum allowed retransmission attempts have been exhausted.

On the other hand, in *type-II hybrid* error control schemes any codewords that could not be successfully decoded are saved, while simultaneously requesting for a retransmission. In case it is not possible to successfully decode the retransmitted codeword, the saved (corrupted) codewords are used to aid in the decoding process. (Note that it is not necessary that the same codeword be retransmitted.) The above process is repeated if the codeword can still not be successfully decoded. We refer the reader to [12,16] for an in-depth discussion of hybrid error control schemes.

For hybrid error control schemes to be effective it is imperative that an optimal mix of FEC and ARQ to be used be determined. The design of hybrid error control schemes is critically dependent on the application being supported as different applications place vastly varying demands on the network. For example, for data transfer, maximum achievable throughput is a key performance measure. On the other hand for packet voice it must be ensured that the end-to-end packet loss probability is below a prespecified threshold and that the tail of the end-to-end packet delay distribution is bounded [21].

For example, let us consider a type-I hybrid ARQ scheme with a block size of n with k information digits, and error correction capability t. We would like to determine the optimal error correction capability, t^*, that should be employed to achieve maximum throughput given an ARQ protocol and FEC coding scheme. The throughput depends on the coding scheme (n, k, t), ARQ protocol, and the channel error probability p. Figure 8 shows the throughput as a function of the error correction capability for various ARQ and FEC schemes. Evaluating closed form expressions for the optimal error correction capability is difficult. However, the asymptotic results can be obtained. For any ARQ protocol, if the product of code length and the channel error rate np is large the optimal error correction capability (for BCH or RS codes) and the maximum throughput are, respectively,

$$t^* \sim np + \mathrm{O}\big(\sqrt{np}\big) \quad \text{and} \tag{2}$$

$$\Lambda\big(p, n, t^*\big) \sim C(1 - cp) + \mathrm{o}(1), \tag{3}$$

c is a constant that depends on the FEC coding scheme being employed. For binary BCH codes $c = m$ and for Reed–Solomon codes $c = 2$. Further, it can be shown that

$$\Lambda_{\mathrm{SR}}(p, n, t) \sim \begin{cases} 0, & \text{if } t < np - \varepsilon np, \\ 1 - cp + \mathrm{o}(1), & \text{if } |np - t| < \varepsilon np, \\ 1 - ct/n, & \text{if } np + \varepsilon np < t. \end{cases}$$

The above asymptotic results indicate that if the np is large and if the error correction capability, t of the FEC scheme used is smaller than np, minimal throughput is obtained. If t is larger than np, the achievable throughput decreases linearly with increase in t. This "threshold" nature of the function $\Lambda(p, n, t)$ can be easily seen (with threshold $t/np = 1$) in figure 8. As expected, the threshold effect is more pronounced as np increases.

The asymptotic results are explained intuitively as follows. Valuable transmission capacity is wasted if the FEC is either too weak or too strong. For optimal operation, the amount of FEC needed for each codeword should be exactly equal to the number of errors in the received codeword. However, since errors occur randomly ideal operation cannot be achieved. An approach to achieve close to ideal operation would involve choosing the error correction capability of np, the expected number of errors in a codeword of size n which is transmitted over a channel with error probability p. As the codeword size increases the *covariance* (covariance of a random variable is defined as the ratio of the variance of the random variable to the square of its mean) in the number of errors in any given codeword decreases. This implies, that the FEC ability provided for each codeword, is more often that not, just enough to correct the errors. This means that close to ideal operation can be obtained as the codeword size increases for any given transmission channel.

We now discuss the implication of the asymptotic results on the design of hybrid error recovery schemes. From the asymptotic results discussed above it follows that a suitably designed error coding scheme can help utilize most of the available channel capacity. However, to achieve the upper bound it is necessary that a sufficiently large codeword size should be chosen. Furthermore, from (2) and figure 8 it can be concluded that to maximize throughput the choice of error correction capability should be conservative, and should always exceed the threshold. Note that the penalty (due to loss in throughput) for choosing t below the optimal is substantially higher than when t exceeds the optimal. It should also be noted that the asymptotically optimal error correction capability does not depend on the ARQ protocol. A look at figure 8 indicates that the difference between t^*_{GBN} and t^*_{SR} is marginal (also note that $t^*_{\mathrm{GBN}} - t^*_{\mathrm{SR}}$ decreases with increasing np). Therefore, *in a hybrid error control scheme the FEC and the ARQ protocols could be designed independently*.

The above observations were made on the basis of the assumption that the objective is to maximize throughput. We must however caution that network protocols (other than error control) and applications often pose conflicting design choices. Some of the factors that govern the choice of the codeword size are: (i) *packetization delay*, (ii) *store and forward delay* at intermediate nodes, and (iii) the *overhead* due to packet headers.

ARQ schemes are used for data transmission since data applications require error free transmission. FEC codes are generally employed as an error control mechanism in real-time applications because they maintain a constant and bounded time delay. Long deep fades cause extremely bursty channel errors. An adaptive error control scheme such as a hybrid ARQ scheme may be well suited for this kind of fluctuating channel.

Original hybrid ARQ schemes, proposed for data transmission, impose unacceptable delays for real-time services. The concept of error control coding with a limited number

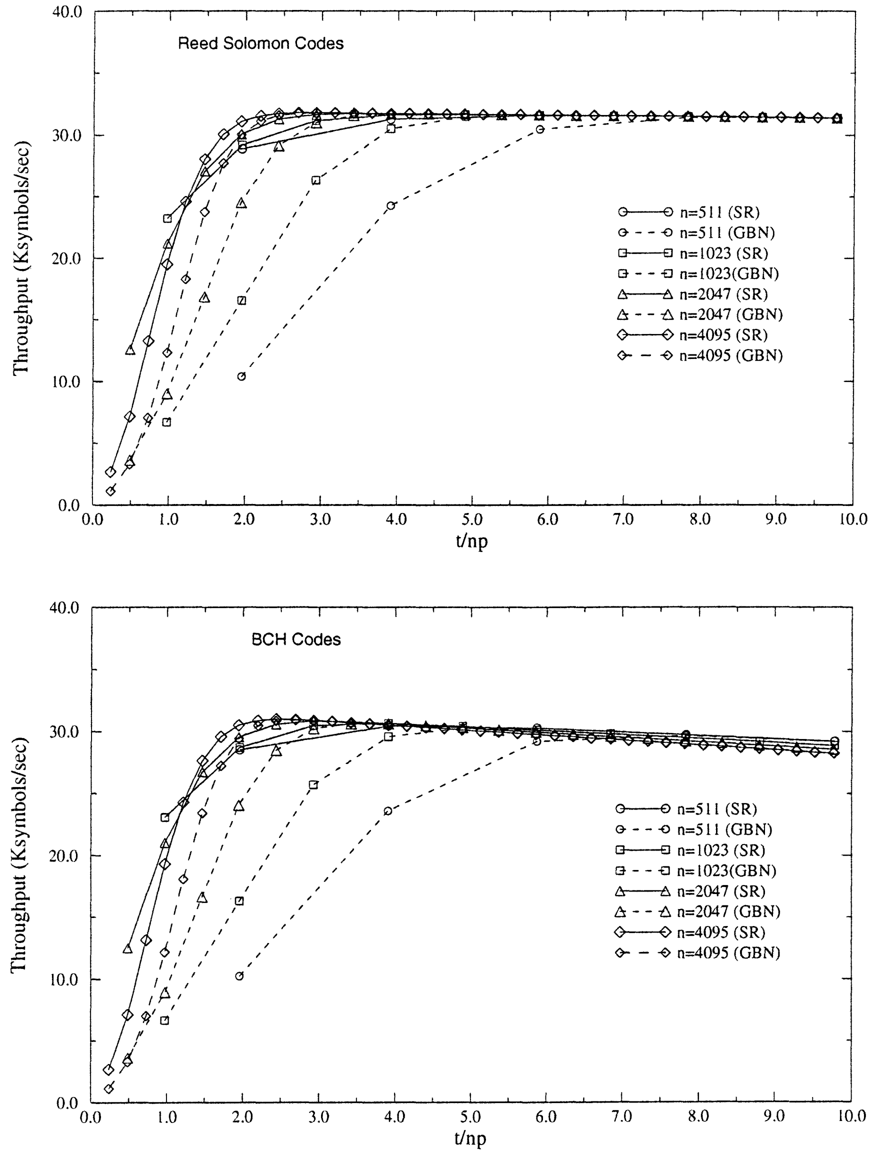


Figure 8. The throughput for the hybrid ARQ scheme using different FEC codes.

of retransmissions is proposed in [20] in order to bound delays (truncated hybrid ARQ). Note that for delay constrained real-time data, the transmitted packets become useless if they do not reach their destination before a predetermined time. Therefore, in truncated hybrid ARQ, the transmitter just drops packets that have not been correctly delivered once their deadline has expired.

5. Case studies

We now present examples from existing and future wireless networks to illustrate the concepts that have been outlined previously. The case studies presented highlight the merits and demerits of various error control methods and discuss the relevance of each for various practical applications. However, before we proceed it is important to review the basic factors that need to be considered when designing optimal error control schemes for various wireless networking applications.

- *Wireless channel characteristics*: There is no universal error control scheme that is optimal for all kinds of channels because of the diversity of wireless channels. An investigation and modeling of the channel characteristics is the first step in the design of an error control scheme.
- *Service requirements*: Different kinds of services have different QoS requirements which impose different requirements on error control, such as BER and delay.
- *Adaptation and graceful degradation*: Wireless channels are time-varying. An effective error control scheme should be capable of adapting to the channel conditions. When the channel conditions are good, less error protection is needed, so that more information can be transmitted. When the channel conditions are poor, more

protection needs to be added. An error control scheme should also consider the quality requirements of sources. Different error protection capabilities are provided to the data based on its importance to the overall quality and its resilience to errors. For example, an ATM cell header needs more protection. So do the I-frames in MPEG video sequences.

- Complexity and availability of off-the-shelf technologies.

5.1. Narrow band systems

The second generation cellular systems are based on digital technology. These systems are currently being deployed. These systems are designed to support low bitrate services using circuit-switched time division multiple access (TDMA) or code division multiple access (CDMA).

5.1.1. Global System for Mobile Communications (GSM)

The Pan-European GSM System uses TDMA (Time Division Multiple Access) with eight timeslots per radio channel [22,28]. Each user transmits periodically in a slot. The slot duration is approximately 0.58 ms which leads to a TDMA frame duration of approximately 4.6 ms. The information is transmitted in bursts at a rate of approximately 271 kbit/s using Gaussian Minimum Shift Keying (GMSK) modulation over a bandwidth of 200 kHz. GSM supports two types of traffic channels: a full rate traffic channel, which carries channel encoded information at a rate of 22.8 kbit/s, and half rate traffic channel at a rate of 11.4 kbit/s. A full rate traffic channel occupies one timeslot. For the half-rate traffic, two connections are mapped onto the same timeslot, transmitting data in alternating frames.

Here we only discuss the error control used in the full rate traffic channel. The interested reader can find more details in [22,28]. For full rate speech transmission, the speech coder delivers 260 bits every 20 ms at a bit rate of 13 kbit/s. The first 50 significant speech bits are encoded using a (53, 50) error detecting block code with the generator polynomial $G(D) = D^3 + D + 1$. The coded 53 bits and the next 132 bits are reordered. 4 tail bits are then added for terminating the convolutional encoder memory. These 189 bits are encoded by a (2, 1, 4) convolutional encoder to give 378 bits. The code uses the generator polynomial $G^{(1)} = 1 + D^3 + D^4$ and $G^{(2)} = 1 + D + D^3 + D^4$. The least significant 78 bits are left unprotected. This results in a total frame length of 456 bits, which corresponds to a bitrate of 22.8 kbit/s.

This frame is then divided into eight 57-bit sub-blocks. The sub-blocks are block diagonally interleaved and then interburst interleaved, resulting in 114-bit bursts that consist of two sub-blocks. A 114-bit burst is transmitted in a time slot with some control information.

User data transmission is also supported at a rate of 9.6 kbit/s, 4.8 kbit/s or 2.4 kbit/s on a full rate traffic channel and at a rate of 4.8 kbit/s or 2.4 kbit/s on a half rate traffic channel. The same half-rate convolutional code used for speech is used but with longer interleaving intervals. A higher layer protocol such as TCP can be employed to gurantee error free delivery using retransmissions.

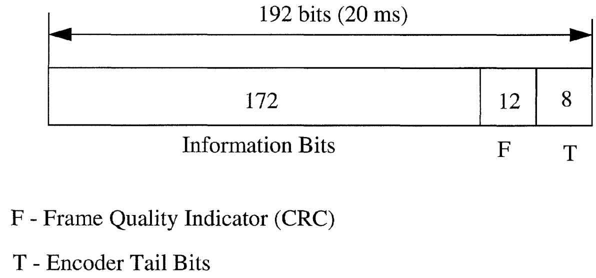


Figure 9. Structure of IS-95 frame.

5.1.2. CDMA based digital cellular networks (IS-95)

The EIA/TIA IS-95 standard is based on CDMA [22, 29,30]. The forward (base-to-mobile) and reverse (mobile-to-base) links use different channel coding and spreading process. For simplicity, of the many data rates supported, we discuss the case of 9.6 kbit/s traffic channels (with a frame period of 20 msec). Each frame contains 172 information bits, 12 CRC bits as frame quality indicator, and 8 tail bits for terminating convolutional encoder memory. Figure 9 shows the frame structure. Before interleaving and spreading, the forward links use a rate $1/2$ convolutional code with the generator functions

$$G^{(1)}(D) = 1 + D + D^2 + D^3 + D^5 + D^7 + D^8$$

and

$$G^{(2)}(D) = 1 + D^2 + D^3 + D^4 + D^8.$$

On the reverse traffic channels, the data is encoded using a rate $1/3$ convolutional code with the generator functions

$$G^{(1)}(D) = 1 + D^2 + D^3 + D^5 + D^6 + D^7 + D^8,$$

$$G^{(2)}(D) = 1 + D^1 + D^3 + D^4 + D^7 + D^8$$

and

$$G^{(3)}(D) = 1 + D^1 + D^2 + D^5 + D^8$$

prior to interleaving and spreading.

Data services are also offered over IS-95 cellular networks. The standard is refered as IS-99 [30]. In the IS-95 standard, the physical layer frame can carry either primary traffic only, or multiplexed primary and secondary traffic. Voice is considered to be primary traffic and user data can be either primary or secondary, determined at call setup time. The error control for data traffic employs two levels of recovery. The lower level recovery protocol, called the radio link protocol (RLP), accomplishes partial error recovery between the mobile and the base station. RLP is based on a NAK-ed ARQ scheme and that allows a maximum of two retransmissions. If the data burst is still in error after two retransmissions, the recovery is left up to the higher layer such as TCP/IP/SNDCF/PPP, to guarantee error free delivery.

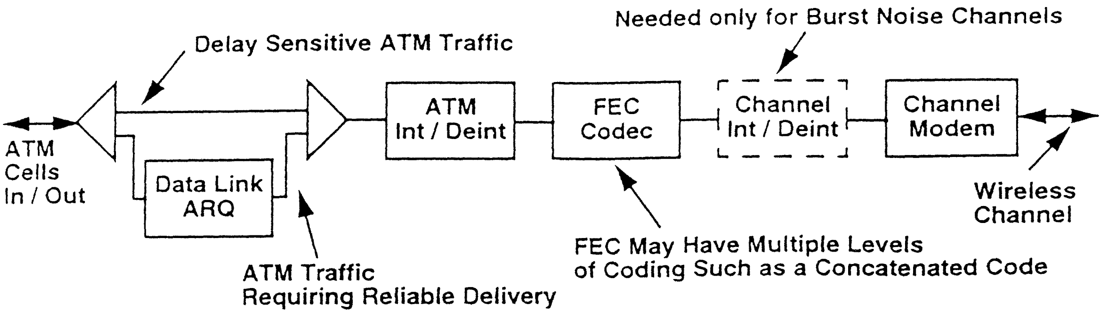


Figure 10. A recommended error control architecture for application of FEC, interleaving, and data link ARQ to provide wireless ATM servicess [33, figure 9].

5.2. Wireless ATM networks

ATM is designed to be a transmission and networking infrastructure for high speed multimedia services (video, voice, data, and others). With the desire to provide ubiquitous access and the acceptance of ATM as a standard for broadband networking applicable to both synchronous and asynchronous services, wireless ATM has been receiving increased attention over the past 2–3 years [1,4,25–27].

ATM is designed for very low error rate and bandwidth-rich media, but wireless channels are time-varying and error prone with limited bandwidth. So, effective and robust data link layer control schemes (error control and flow control) are necessary to support ATM over wireless channels. Below we briefly introduce some of the error control schemes proposed in the recent literature on wireless ATM services.

In [7], an error control architecture is recommended to avoid performance degradation when using ATM over wireless links. The principal components of the proposed architecture and the functions are as follows:

- Channel interleaving to randomize burst errors among different cells.
- FEC to reduce channel error rate. The Reed–Solomon/Viterbi concatenated code is used, it is a very powerful FEC code with reasonable complexity using currently available VLSI decoder chips.
- ATM interleaving to randomize the error bursts out of the FEC decoder within one cell. The cell headers (five bytes each) are interleaved with the data stream so that no more than a single error from a burst error appears in any cell header. Recall that in ATM the cell header is capable of correcting a single bit error.
- Data Link ARQ to provide reliable transmission for applications that require reliable delivery. Data link protocol parameters can be determined in a straightforward manner given the channel characteristics. The Go-Back-N protocol may be used on relatively good channels, whereas Selective Repeat protocol is preferred over severely impaired channels.

In [15], a method for reducing the cell loss probability by interleaving the header and the payload bits over a wireless ATM link is introduced. There exist two distinct methods for performing interleaving in an ATM environment:

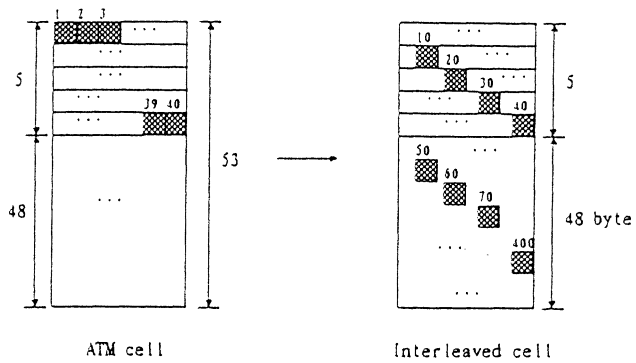


Figure 11. One method of cell unit interleaving method [15, figure 1].

- interleaving headers of multiple cells (block unit interleaving), and
- spreading each bit of a header over the payload field within the cell (cell unit interleaving).

The block unit interleaving method is very effective for burst errors of length up to the number of bits equal to the number of cells in one block unit. This incurs a large processing delay for interleaving. The cell unit interleaving distributes all of the 40 header bits of a cell at 10 bit intervals across the cell. The method is effective for burst errors shorter than 11 bits by transforming burst errors into randomly distributed single-bit errors which can be easily corrected by the ATM header. Even if long bit errors occur, the burst errors will not affect other cells because the interleaving is done within a single cell. Cell unit interleaving also incurs a shorter processing delay. Simulation results verified that the cell unit interleaving scheme reduces cell loss probability effectively and is easily applicable to PDH, SDH and cell-based physical layers. It is also more robust than block unit interleaving against unexpected long burst errors.

A straightforward approach to evolve from ATM to wireless ATM is to use a standard ATM cell for network level functions, while a wireless header-tailer is added on the radio link for wireless channel specific protocol layers – MAC (Medium Access Control), DLC (Data Link Control) and the network control layer [33]. DLC layer is used to provide increased error protection. Service class specific retransmission strategies are used to improve error recovery with a minimal decrease in the wireless channel utilization.

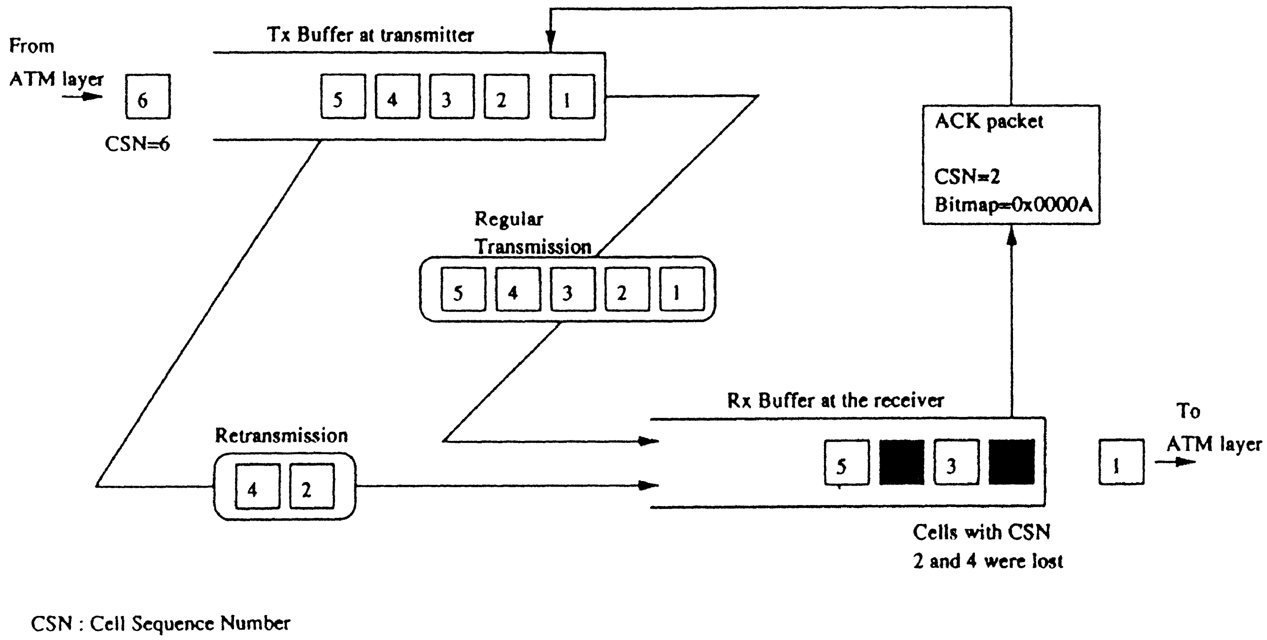


Figure 12. CBR DLC: Operation of FIFO and transssfer of cells to the ATM layer [33, figure 4].

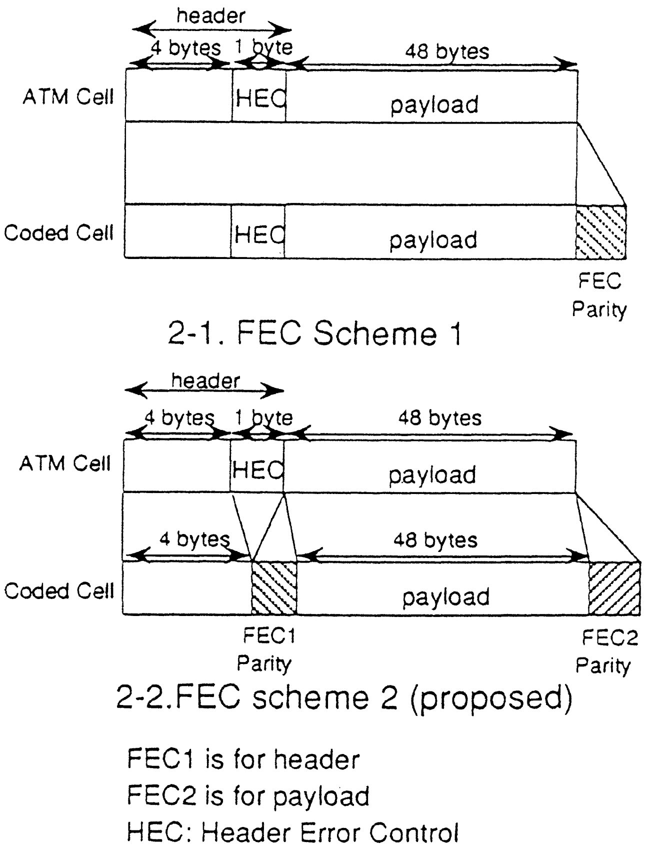


Figure 13. ATM cell and coded cell format [2, figure 2].

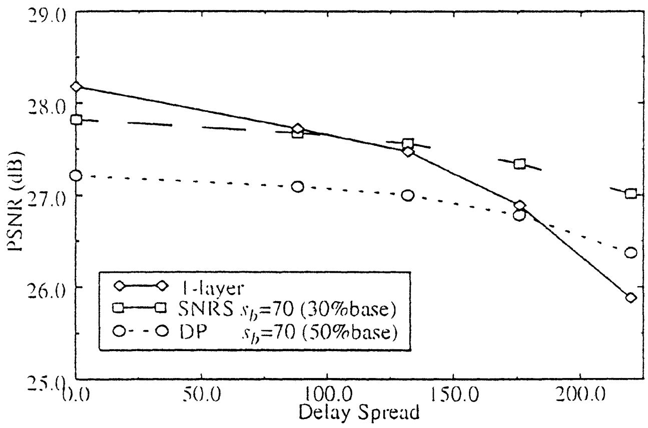


Figure 14. Optimal quality for 1- and 2-layer schemes (Flower Garden) [6, figure 7].

For ABR (available bit rate) services, DLC uses the traditional ARQ procedure as there are no delay bounds. For CBR (constant bit rate) and VBR (variable bit rate), source retransmissions are bounded to a limit specified by the application at VC (virtual connection) setup time. For the CBR service, to maintain a constant bit rate channel, the retransmissions need to use additional bandwidth so as not to impair the throughput of the CBR connection.

ATM employs Header Error Correction (HEC) to protect the ATM cell header from single-bit errors. However, wireless ATM requires a more powerful scheme to improve BER performance. In [2], a new FEC scheme is proposed that modifies the cell structure of ATM cells to improve error correction ability. Two different FEC codes, one for the header, which is a powerful coding gain code, and the other for the payload, which is a higher coding rate code, are adopted. The coded cell is modified into a wireless ATM cell in which the HEC is discarded and two FECs are added. After replacing the HEC in the ATM cells with the FECs, the wireless ATM cells are transmitted. After error correction at the receiver, the FECs are discarded and a new HEC is generated to regenerate the original ATM cell stream.

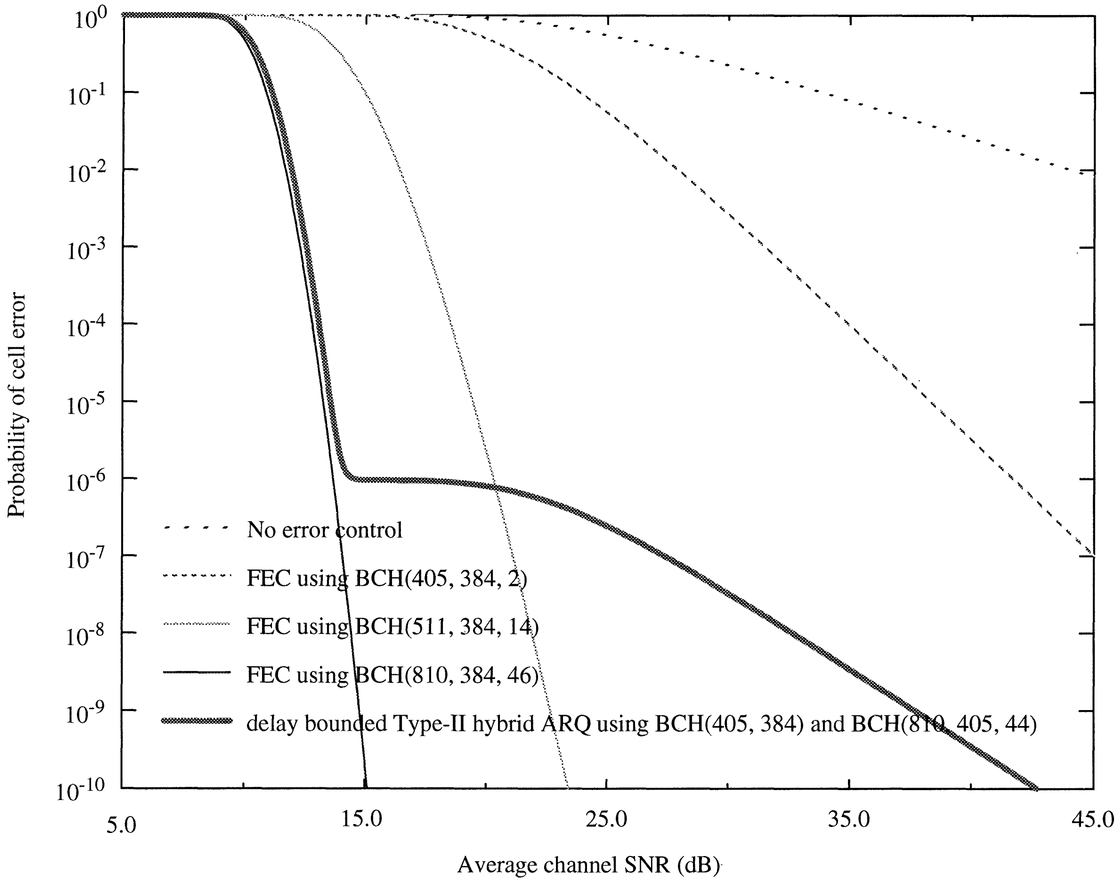


Figure 15. The average PSNR of the reconstructed video sequence under different error control schemes [17, figure 4].

One major service class supported by ATM constitutes video-related services which usually require high throughput, low bit error rate and are delay sensitive. Because of the time-varying property of the wireless link, layered video source coding and unequal error protection schemes have received great attention in recent years.

Layered coding and unequal error protection schemes raise two questions. One is how to code the source information into different priority layers, and the other is how to choose appropriate error protection for different priority layers. An FEC-based error control scheme in combination with 2-layer video coding techniques is used to transport the MPEG-2 video over an indoor broadband ad hoc wireless ATM LAN is proposed in [5]. Two models for the indoor TDMA wireless channel are presented. One is the random loss model, where the wireless channel is characterized by uncorrelated bit errors introduced by random noise and interference components. The other is the multipath loss model, where the transmitted signal undergoes impairment due to multipath fading, shadowing and co-channel interference. Most wireless channels can be modeled by a combination of these two models. Error control on the wireless link is achieved by means of a channel encoder–decoder. RS encoding–decoding was chosen because it leads to minimal overhead and the commercial availability of 80 Mb/s encoders–decoders. The FEC-based scheme includes three possible levels [3]:

- *Bit-level FEC*: This is done at the physical layer, typically in hardware, by means of a DSP chip or a specific IC. For a bandwidth limited channel, trellis coded modulation with Viterbi decoding is used. If the channel is not band limited, block or convolutional encoding are employed.
- *Byte-level FEC*: This is done on a per-packet basis. Traditionally, every packet carries a CRC field for error detection only. Recently, because of more powerful processing abilities, use of this field for error correction is also possible. The authors adopted the Reed–Solomon(RS) encoder to process symbols where a symbol is a group of *m* bits. The RS encoder processes N data symbols to generate $2t$ symbols, where t is the number of symbols that can be corrected by a RS encoder.
- *Cell-level FEC*: This is done by allocating some redundant cells for error correction. In the case of cells that cannot be corrected by bit-level or byte-level FEC, the redundant cells, together with the correctly received cells, are used to recover the lost cells without retransmissions. For instance, the cell sequence numbers are used to detect the location of lost cells. *k* RS symbols are then sufficient to recover from *k* errored symbols. Thus, up to *s* lost cells can be recovered by the *s* redundant cells using a RS codec or another maximum distance

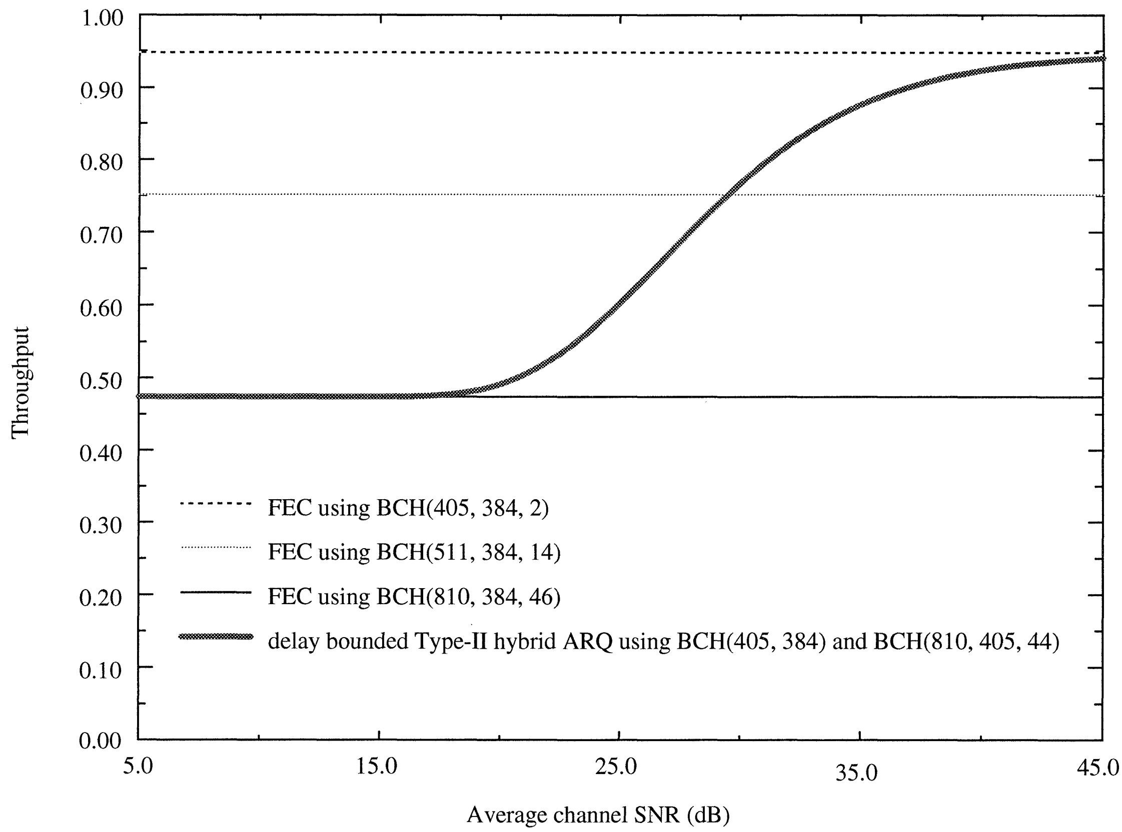


Figure 16. The throughputs of the system under the different error control schemes [17, figure 5].

separable code. An alternative to cell-level coding is to use per-cell coding with interleaving at the cost of delay due to interleaving.

In order to achieve best possible video quality over wireless ATM, scalable or layered source coding is necessary. Several 2-layer coding techniques have been investigated in the literature [5].

- *Data Partitioning* (DP), which divides encoded DCT coefficients into two groups: low frequency components into a base layer with high priority (HP) and high frequency components into an enhancement layer with low priority (LP).
- *SNR Scalability* (SNRS), which first uses a coarse quantizer to encode the DCT coefficients to create a base layer with high priority (HP) and then uses a fine quantizer on the errors to create an enhancement layer with low priority (LP).

Simulation results based on the error characteristics of each model showed that byte-level coding was appropriate for random losses while cell-level coding provides protection against multipath channel losses. For both random and multipath channel models, 2-layer scalable MPEG-2 outperforms 1-layer MPEG-2, and 2-layer coding using SNR Scalability outperforms Data Partitioning at a cost of complexity [6].

In [17], a delay-bounded type-II hybrid ARQ scheme using BCH codes is investigated for delivering H.263 coded video over indoor wireless ATM link. Hybrid ARQ makes use of both FEC and ARQ to achieve near optimal throughput and reliability. Simulation results show that significant improvement in video quality can be achieved.

6. Conclusions

We investigated the issue of error control in wireless communication networks. We examined the alternatives available for providing a reliable end-to-end communication in wireless environments, and discussed their tradeoffs. The channel conditions are different in different wireless environments such as indoor and outdoor, high mobile speed and low mobile speed. QoS requirements are also dependent on the applications. An overall optimal solution does not exist. A solution should be based on the application, the environment and the QoS requirements to design the error control scheme and achieve the best solution.

References

[1] A. Acampora, Wireless ATM: A perspective on issues and prospects, IEEE Personal Communications (August 1996).

[2] S. Aikawa et al., Forward error correction schemes for wireless ATM systems, in: *Proc. IEEE ICC ’96* (1996) p. 454.

[3] E. Ayanoglu et al., AIRMAIL: A link-layer protocol for wireless networks, Wireless Networks 1(1) (February 1995).

[4] E. Ayanoglu et al., Wireless ATM: Limits, Challenges, and Proposals, IEEE Personal Communications (August 1996).

[5] E. Ayanoglu et al., Forward error control for MPEG-2 video transport in a wireless ATM LAN, Mobile Networks and Applications (September 1996).

[6] E. Ayanoglu et al., Forward error control for MPEG-2 video transport in a wireless ATM LAN, in: *Proc. ICIP' 96* (1996).

[7] J.B. Cain et al., A recommended error control architecture for ATM networks with wireless links, IEEE Journal on Selected Areas in Communications 15(1) (January 1997).

[8] G. Clark and J. Cain, *Error-Correction Coding for Digital Communications* (Plenum Press, New York, 1981).

[9] G. Clark, J. Cain and J. Geist, Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding, IEEE Transactions on Information Theory 25 (1979) 97–110.

[10] Group Special Mobile (GSM) Recommendation (April 1988).

[11] D. Haccoun and G. Begin, High-rate punctured convolutional codes for Viterbi and sequential decoding, IEEE Transactions on Communications 31(11) (1989) 1113–1125.

[12] J. Hagenauer, Rate-compatible punctured convolutional codes (RCPC codes) and their applications, IEEE Transactions on Communications 36 (1988) 389–400.

[13] S. Kallel and D. Haccoun, Generalized type-II hybrid ARQ scheme using punctured convolutional coding, IEEE Transactions on Communications 38(11) (1990) 1938–1846.

[14] L. Lee, Concatenated coding systems employing a unit-memory convolutional code and a byte-oriented decoding algorithm, IEEE Transactions on Communications 25(10) (1977) 1064–1074.

[15] S. Lim et al., Cell loss reduction by cell unit interleaving in wireless ATM Networks, in: *IEEE ICC '96* (1996) p. 449.

[16] S. Lin and D.J. Costello Jr, *Error Control Coding: Fundamentals and Applications* (Prentice-Hall, Englewood Cliffs, NJ, 1983).

[17] H. Liu and M. El Zarki, Delay bounded type-II hybrid ARQ for video transmission over wireless channel, in: *Proc. Conference on Information Sciences and Systems*, Princeton, NJ (March 1996).

[18] L. Lugand, D. Costello and R. Deng, Parity retransmission hybrid ARQ using rate 1/2 convolutional codes on a nonstationary channel, IEEE Transactions on Communications 37(4) (1989) 755–765.

[19] A. Michelson and A. Levesque, *Error-Control Techniques for Digital Communication* (Wiley, New York, 1985).

[20] M. Nakamura and T. Kodama, Performance evaluation for ARQ schemes in power and/or bandwidth limited systems, Transactions of IEICE 72(5) (1989) 491–501.

[21] R.O. Onvural, *Asynchronous Transfer Mode Networks: Performance Issues* (Artech House, 1993).

[22] J.E. Padgett, C.G. Gunther and T. Hattori, Overview of wireless personal communications, IEEE Communications Magazine 33(1) (1995) 28–41.

[23] J.G. Proakis, *Digital Communications* (McGraw-Hill, 1989).

[24] D. Raychaudhuri, Wireless ATM networks: Architecture, system design and prototyping, IEEE Personal Communications (August 1996).

[25] Special Issue: Wireless ATM, IEEE Personal Communications (August 1996).

[26] Special Issue: Wireless ATM, Mobile Networking and Applications (September 1996).

[27] Special Issue: Wireless ATM, IEEE Journal on Selected Areas in Communications (January 1997).

[28] R. Steele, *Mobile Radio Communications* (Pentech Press, London, 1992).

[29] TIA/EIA IS-95, Mobile Station-Base Station Compatibility Standard for Dual Mode Wide-Band Spread-Spectrum Cellular Systems, Telecommunications Industry Association (1992).

[30] TIA/EIA IS-99, Data Services Option Standard for Wideband Spread Spectrum Digital Cellular System, Telecommunications Industry Association (September 1994).

[31] Y. Wang and S. Lin, A modified Selective-Repeat type-II hybrid ARQ system and its performance analysis, IEEE Transactions on Communications 31(5) (1983) 124–133.

[32] S.B. Wicker, Reed–Solomon error control coding for Rayleigh fading channels with feedback, IEEE Transactions on Vehicular Technology 41(2) (1992) 124–133.

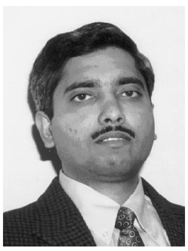
[33] H. Xie et al., Data link control for wireless ATM access channels, in: *Proc. ICUPC '95* (1995).

[34] M. Zorzi and R.R. Rao, Capture and retransmission control in mobile radio, IEEE Journal on Selected Areas in Communications 12(8) (1994) 1289–1298.

Hang Liu received the B.S. from Tianjin University, China, in 1985, and the M.S. from the University of New Orleans, New Orleans, LA, in 1992. He is currently a Ph.D. candidate in the Department of Electrical Engineering, University of Pennsylvania. His research interests include video compression and communications, wireless networking, ATM based networks and digital signal processing.
E-mail: hliu@ee.upenn.edu

Hairuo Ma received his B.S. from Tsinghua University, Beijing, P.R. China, and his M.S. from the University of Pennsylvania, both in electrical engineering, in 1993 and 1995, respectively. In 1996, he was a software engineer at DTI, Maryland. Currently, he is a Ph.D. candidate in the Department of Electrical Engineering, Univ. of Penn. His research interests include wireless networking, wireless ATM, broadband service to the home and multiresolution transmission.
E-mail: hairuo@ee.upenn.edu

Magda El Zarki received the B.E.E. degree from Cairo University, Egypt, in 1979, and the M.S. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, in 1981 and 1987, respectively. She worked from 1981 to 1983 as a communication network planner in the Department of International Telecommunications at Citibank in New York. She joined Columbia University in 1983 as a research assistant in the Computer Communications Research Laboratory where she was involved in the design and development of an integrated local area network testbed called MAGNET. In 1988 she joined the faculty of the Department of Electrical Engineering of the University of Pennsylvania, teaching courses and conducting research in the field of telecommunications, where she currently serves as an associate professor. She also holds a secondary appointment in the Department of Computer and Information Sciences. In January 1993, she was appointed as a part-time professor of Telecommunication Networks in the Faculty of Electrical Engineering at Delft University of Technology, in Delft, The Netherlands. Dr. El Zarki is a member of the ACM, IEEE and Sigmaa Xi. She is actively involved in many of their sponsored conferences and journals. She was the Technical Program Chair of IEEE INFOCOM '94.
E-mail: magda@ee.upenn.edu



Sanjay Gupta received his Ph.D. in systems in 1993 and M.S. degree in electrical engineering in 1989, both from the University of Pennsylvaniaa. He received his B. Tech. degree in electrical engineering in 1988 from the Indian Institute of Technology, Kanpur. He joined the Department of Electrical and Computer Engineering at Illinois Institute of Technology in 1993 as an Assiatant Professor. He was a visiting research scholar at Hong Kong University of Science and Technology during the summer of 1995 and 1996. He is currently employed at Motorola, where he is working on UMTS and GSM. His oher research interests include wireless local area networks, ATM based broadband networks, management and control of communication networks, and stochastic modeling and simulation. He has served as a Technical Program Committee member for the IEEE Infocom conferences since 1993 and the IEEE Vehicular Technology Conference 1995.

E-mail: guptasn@cig.mot.com