

Applications of Error-Control Coding

Daniel J. Costello, Jr., *Fellow, IEEE*, Joachim Hagenauer, *Fellow, IEEE*,
Hideki Imai, *Fellow, IEEE*, and Stephen B. Wicker, *Senior Member, IEEE*

(Invited Paper)

Abstract—An overview of the many practical applications of channel coding theory in the past 50 years is presented. The following application areas are included: deep space communication, satellite communication, data transmission, data storage, mobile communication, file transfer, and digital audio/video transmission. Examples, both historical and current, are given that typify the different approaches used in each application area. Although no attempt is made to be comprehensive in our coverage, the examples chosen clearly illustrate the richness, variety, and importance of error-control coding methods in modern digital applications.

Index Terms—Block codes, channel coding, convolutional codes, error-control coding.

I. INTRODUCTION

WITH his 1948 paper, “A Mathematical Theory of Communication,” Shannon [1] stimulated a body of research that has evolved into the two modern fields of Information Theory and Coding Theory. The fundamental philosophical contribution of [1] was the formal application of probability theory to the study and analysis of communication systems. The theoretical contribution of Shannon’s work in the area of channel coding was a useful definition of “information” and several “channel coding theorems” which gave explicit upper bounds, called the channel capacity, on the rate at which “information” could be transmitted reliably on a given communication channel.

In the context of this paper, the result of primary interest is the “noisy channel coding theorem for continuous channels with average power limitations.” This theorem states that the capacity C of a bandlimited additive white Gaussian noise (AWGN) channel with bandwidth W , a channel model that approximately represents many practical digital communication and storage systems, is given by

$$C = W \log_2(1 + E_s/N_0) \text{ bits per second (bps)} \quad (1)$$

Manuscript received March 2, 1998; revised June 1, 1998.

D. J. Costello, Jr. is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (phone: +1-219-631-5480, fax: +1-219-631-4393, e-mail: Daniel.J.Costello.2@nd.edu).

J. Hagenauer is with the Institute of Communications Engineering (LNT), Munich University of Technology (TUM), 80290 Munich, Germany (phone.: +49-89-28923467, fax: +49-89-28923490, e-mail: Hag@LNT.E-Technik.TU-Muenchen.DE).

H. Imai is with the Institute of Industrial Science, University of Tokyo, Minato-ku Tokyo, 106-8558, Japan (phone: +81-3-3402-6231, fax: +81-3-3402-6425, e-mail: imai@iis.u-tokyo.ac.jp).

S. B. Wicker is with the School of Electrical Engineering, Cornell University, Ithaca, NY 14853 USA (phone: +1-607-255-8817, fax: +1-607-255-9072, e-mail: wicker@ee.cornell.edu).

Publisher Item Identifier S 0018-9448(98)06086-6.

where we assume perfect Nyquist signaling, E_s is the average signal energy in each signaling interval of duration $T = 1/W$, and $N_0/2$ is the two-sided noise power spectral density. (In this formulation of the capacity theorem, one quadrature (two-dimensional) signal is transmitted in each T -second signaling interval, and the nominal channel bandwidth $W = 1/T$. See Wozencraft and Jacobs [2] for a more complete discussion of the concept of channel bandwidth.) The proof of the theorem demonstrates that for any transmission rate R less than or equal to the channel capacity C , there exists a coding scheme that achieves an arbitrarily small probability of error; conversely, if R is greater than C , no coding scheme can achieve reliable performance. However, since this is an existence theorem, it gives no guidance as to how to find appropriate coding schemes or how complex they may be to implement.

Beginning with the work of Hamming, which was published in 1950 [3] but was already known to Shannon in 1948, many communication engineers and coding theorists have developed numerous schemes for a variety of applications in an attempt to achieve performance close to what was promised by Shannon with reasonable implementation complexity. In this paper, we will survey the progress made in applying error-control coding techniques to digital communication and storage systems over the past 50 years and see that great advances have occurred in designing practical systems that narrow the gap between real system performance and channel capacity. In particular, we will focus on applications in six areas: space and satellite communications (Section II), data transmission (Section III), data storage (Section IV), digital audio/video transmission (Section V), mobile communications (Section VI), and file transfer (Section VII). Included among the applications covered in this survey are the Consultative Committee on Space Data Systems (CCSDS) standard coding scheme for space and satellite communications, trellis coding standards for high-speed data modems, the Reed–Solomon coding scheme used in compact discs, coding standards for mobile cellular communication, and the CRC codes used in HDLC protocols. The reader may wish to consult the paper published in 1974 by Jacobs [4], which reviewed applications of error-control coding over the first 25 years after the publication of Shannon’s paper, to get an appreciation for the accelerated rate at which coding techniques have been applied to real systems in recent years.

The result in (1) can be put into a form more useful for the present discussion by introducing the parameter η , called the spectral (or bandwidth) efficiency. That is, η represents the

Spectral Efficiency, η , versus E_b/N_0

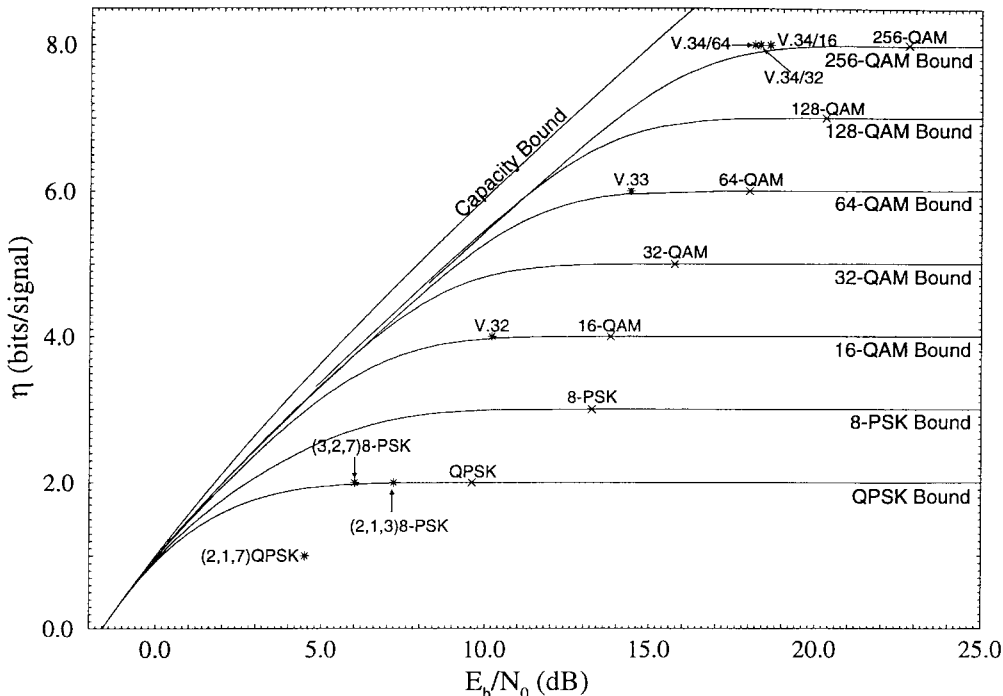


Fig. 1. Capacity curves and the performance of several coding schemes for data transmission applications.

average number of information bits transmitted per signaling interval of duration T seconds. Then

$$0 < \eta < C/W \tag{2}$$

and

$$E_s/N_0 = \eta E_b/N_0 \tag{3}$$

where E_b is the average energy per information bit. Substituting the above relations into (1) and performing some minor manipulations yields

$$E_b/N_0 > (2^\eta - 1)/\eta \tag{4}$$

which relates the spectral efficiency η to the signal-to-noise ratio (SNR), represented by E_b/N_0 . The bound of (4) expresses the fundamental tradeoff between the spectral efficiency η and the SNR E_b/N_0 . That is, increased spectral efficiency can be reliably achieved only with a corresponding increase in the minimum required SNR. Conversely, the minimum required SNR can be reduced only by decreasing the spectral efficiency of the system.

The bound of (4) is shown plotted in Fig. 1 and is labeled the capacity bound. This curve represents the absolute best performance possible for a communication system on the AWGN channel. The performance of a particular system relative to the capacity bound may be interpreted in two distinct ways.

First, the capacity bound may be interpreted as giving the minimum SNR required to achieve a specific spectral efficiency with an arbitrarily small probability of error. For example, if one wants to transmit $\eta = 1$ information bit per signal, then there exists a coding scheme that operates reliably with an $E_b/N_0 = 0$ dB. Conversely, any coding

scheme, no matter how complex, sending $\eta = 1$ information bit per signal with an E_b/N_0 less than 0 dB will be unreliable. This interpretation indicates the maximum power reduction available using appropriate coding schemes compared to an uncoded system. As an illustration, consider that an uncoded quadrature phase-shift keyed (QPSK) system with coherent detection has a spectral efficiency of $\eta = 2$ and achieves a bit-error rate (BER) of 10^{-5} at an $E_b/N_0 = 9.6$ dB. Since coding can provide essentially error-free communication at the same spectral efficiency and an $E_b/N_0 = 1.8$ dB, we say that a maximum power (or coding) gain of 7.8 dB is available in this case.

Second, the capacity bound may be interpreted as giving the maximum spectral efficiency at which a system may operate reliably for a fixed SNR. For example, if an $E_b/N_0 = 0$ dB is available, then there exists a coding scheme that operates reliably with a bandwidth efficiency of $\eta = 1$ information bit per signal. Conversely, any coding scheme, no matter how complex, sending more than $\eta = 1$ information bit per signal will be unreliable if $E_b/N_0 = 0$ dB. This interpretation indicates the maximum spectral efficiency increase available using appropriate coding schemes compared to an uncoded system. As an illustration, again consider an uncoded QPSK system with coherent detection. Since a coded system operating at an $E_b/N_0 = 9.6$ dB can provide reliable communication at a spectral efficiency of $\eta = 5.7$ information bits per signal, we say that a maximum spectral efficiency (or rate) gain of 3.7 bits per signal is available in this case.

The capacity bound assumes that, for a given spectral efficiency, one is free to choose the signaling (modulation) scheme which results in the best possible performance. However, in

real communication systems, there are many practical considerations that come into play in choosing a modulation scheme. For example, satellite communication systems that use nonlinear traveling-wave tube amplifiers (TWTA's) require constant envelope signaling such as M -ary phase-shift keying (M -PSK) in order to avoid signal distortion. Also, for data transmission over voice-grade telephone channels, the intersymbol interference caused by bandwidth limitations necessitates the use of large signal constellations that employ a combination of amplitude and phase modulation (AM/PM) to achieve high data rates. It is, therefore, instructive to compute the maximum spectral efficiency η required to achieve reliable communication given a particular modulation scheme and SNR.

For the discrete-input, continuous-output, memoryless AWGN channel with M -ary phase-shift-keyed (M -PSK) or quadrature amplitude modulation (M -QAM) modulation and assuming equiprobable signaling, the capacity bound becomes [2]

$$\eta^* < \log_2(M) - (1/M) \sum_{i=0}^{M-1} E \left\{ \log_2 \sum_{j=0}^{M-1} \exp[-(|a^i + n - a^j|^2 - |n|^2)/N_0] \right\} \quad (5)$$

where a^i is a channel signal, n is a Gaussian distributed noise random variable with mean 0 and variance $N_0/2$, and E is the expectation operator. The bound of (5) is plotted in Fig. 1 for equiprobable QPSK, 8-PSK, 16-QAM, 32-QAM, 64-QAM, 128-QAM, and 256-QAM modulation. (For M -QAM signal sets with large M , nonequiprobable signaling, called signal shaping, can improve performance by as much as 1.53 dB compared to equiprobable signaling [5].)

For a specified signaling method and SNR, the bound of (5) represents the maximum spectral efficiency required to achieve reliable communication. For example, a QPSK system operating with an $E_b/N_0 = 1.64$ dB can transmit a maximum of $\eta = 1.5$ bits per signal. This is 0.44 bit per signal less than an ideal system without any modulation constraints. Alternatively, to send $\eta = 1.5$ information bits per signal using QPSK modulation requires a minimum $E_b/N_0 = 1.64$ dB. This is 0.76 dB more than an ideal system without any modulation constraints.

In the sections that follow, we will have occasion to compare some of the coding systems discussed to the capacity curves in Fig. 1 in order to assess their relative power and/or spectral efficiency, at least in the cases where AWGN is the primary signal impairment. In other cases, such as mobile communications, where signal fading is the major source of errors, and data storage, where media contamination causes most of the problems, a clear view of optimum performance is not as readily available.

II. APPLICATIONS TO SPACE AND SATELLITE COMMUNICATIONS

Most of the early work in coding theory was directed at the low spectral efficiency, or power-limited, portion of the capacity curve. This was principally due to two factors. First,

many early applications of coding were developed for the National Aeronautics and Space Administration (NASA) and the European Space Agency (ESA) deep-space and satellite communication systems, where power was very expensive and bandwidth was plentiful. Second, no practical coding schemes existed that could provide meaningful power gains at higher spectral efficiencies. Thus our paper begins with a survey of applications of error-control coding to space and satellite communication systems.

The deep-space channel turned out to be the perfect link on which to first demonstrate the power of coding. There were several reasons for this, most notably those listed below.

- 1) The deep-space channel is almost exactly modeled as the memoryless AWGN channel that formed the basis for Shannon's noisy channel coding theorem. Thus all the theoretical and simulation studies conducted for this channel carried over almost exactly into practice.
- 2) Plenty of bandwidth is available on the deep-space channel, thus allowing the use of the low-spectral-efficiency codes and binary-modulation schemes that were most studied and best understood at the time.
- 3) Because of the large transmission distances involved, which caused severe signal attenuation, powerful, low-rate codes, with complex decoding methods, were required, resulting in very low data rates. However, since a deep-space mission is, by nature, a very time-consuming process, the low data rates realized in practice did not present a problem.
- 4) A deep-space mission is also, by nature, a very expensive undertaking, and thus the additional cost of developing and implementing complex encoding and decoding solutions can be tolerated, especially since each decibel of coding gain realized resulted in an overall savings of about \$1 000 000 (in the 1960's) in transmitting and receiving equipment.

Thus it is no surprise that Massey, in a recent paper [6], called deep-space communication and coding a "marriage made in heaven."

As a starting point to understand the gains afforded by coding on the deep-space channel, we consider an uncoded BPSK system with coherent detection. (Throughout this section we assume BPSK modulation, which transmits uncoded information at a rate of 1.0 information bit per signal. If coding is used, the code rate r , in information bits per BPSK signal, then represents the spectral efficiency of the coded system.) Simulation results and analytical calculations have shown that uncoded BPSK achieves a BER of 10^{-5} , considered "reliable" in many applications, at an $E_b/N_0 = 9.6$ dB. This point is plotted in Fig. 2 with the label BPSK. (All the points in Figs. 1 and 2 are plotted for a BER of 10^{-5} . Actual BER requirements vary from system to system in deep-space applications depending on several factors, such as the nature and sensitivity of the data and whether it has been compressed on the spacecraft prior to transmission.)

From the capacity curve in Fig. 2, it can be seen that the minimum E_b/N_0 required to achieve error-free communication at a code rate $r = 1/2$ bit per signal is 0.0 dB, and

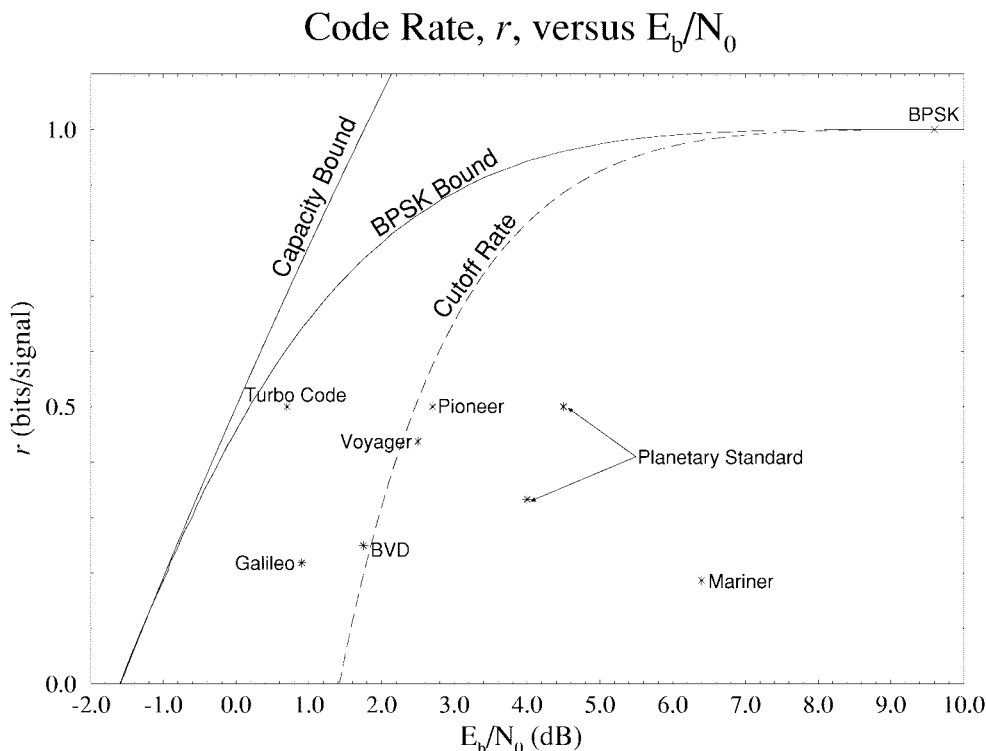


Fig. 2. Capacity and cutoff rate curves and the performance of several coding schemes for deep-space applications.

thus a power savings of 9.6 dB is theoretically possible with an appropriate rate $r = 1/2$ coding scheme. Looking at the BPSK capacity curve, however, reveals that to achieve a rate $r = 1/2$ with BPSK modulation requires only a slightly larger $E_b/N_0 = 0.2$ dB. Thus for code rates $r = 1/2$ or less, very little in potential coding gain is sacrificed by using BPSK modulation. This, combined with the difficulty in coherently detecting signal sets with more than two points and the relative abundance of bandwidth on the deep-space channel, resulted in the choice of BPSK modulation with code rates $r = 1/2$ bit/signal or below for deep-space communication.

One of the earliest attempts to improve on the performance of uncoded BPSK was the use of a rate $r = 6/32$ biorthogonal, Reed–Muller block code, also referred to as the $(32, 6)$ RM code. This code was used on the 1969 Mariner and later Viking Mars missions in conjunction with BPSK modulation and soft-decision maximum-likelihood decoding. The code consisted of 64 codewords, each 32 bits long, with a minimum Hamming distance between codewords of $d_{\min} = 16$. The 64 codewords can be viewed as a set of 32 orthogonal vectors in 32-dimensional space, plus the complements of these 32 vectors, and thus the name “biorthogonal.” Full soft-decision maximum-likelihood decoding (decoding using unquantized demodulator outputs) was achieved by using a correlation decoder, based on the Hadamard transform, developed by Green at the Jet Propulsion Laboratory (JPL) that subsequently became known as the “Green Machine” [7].

The Mariner system had code rate $r = 6/32 = 0.1875$ bit/signal and it achieved a BER of 10^{-5} with an $E_b/N_0 = 6.4$ dB. (Note that, since BPSK modulation is assumed, each BPSK signal represents one code bit, and thus the code rate r

is equivalent to the spectral efficiency.) This point is plotted in Fig. 2 with the label “Mariner.” From Fig. 2, it is seen that the Mariner code requires 3.2 dB less power than uncoded BPSK for the same BER, but it requires more than five times the bandwidth and is still 7.5 dB away from the BPSK capacity curve at the same spectral efficiency. It is important to note that even with its significant bandwidth expansion, the coding gain actually achieved by the Mariner code was rather modest. This is due to the fact that this code, as is typical of block codes in general, has a relatively large number of nearest neighbor codewords, thus substantially reducing the available coding gain at moderate BER’s. (At lower BER’s, a coding gain of up to 4.8 dB is achievable, but this is reduced to 3.2 dB at a BER of 10^{-5} by the code’s 62 nearest neighbors.) In fact, most of the coding gain achieved by the Mariner code was due to the extra 2–3 dB obtained by using full soft-decision decoding, a lesson that has carried over to almost all practical coding implementations where coding gain is a primary consideration.

A significant advance in the application of coding to deep-space communication systems occurred later in the 1960’s with the invention of sequential decoding [8] for convolutional codes and its subsequent refinement [9]. It was now possible to use powerful long-constraint-length convolutional codes with soft-decision decoding. Thus for the first time, practical communication systems were capable of achieving substantial coding gains over uncoded transmission.

Sequential decoding was first used in 1968 on an “experimental” basis. (This was actually a deliberate stratagem to circumvent lengthy NASA qualification procedures [6].) The Pioneer 9 solar orbit space mission used a modified version of a rate $r = 1/2$ systematic convolutional code originally

constructed by Lin and Lyne [10], but the coding scheme was changed for subsequent missions. (A convolutional code is said to be in systematic form if the information sequence appears unchanged as one of the encoded sequences.) It is interesting to note that, even though the Mariner coding system was designed first, the Pioneer 9 was actually launched earlier, and thus the Lin–Lyne code was the first to fly in space. The Pioneer 10 Jupiter fly-by mission and the Pioneer 11 Saturn fly-by mission in 1972 and 1973, respectively, both used a rate $r = 1/2$, constraint length $K = 32$, i.e., a $(2, 1, 32)$ nonsystematic, quick-look-in (QLI) convolutional code constructed by Massey and Costello [11]. The two 32-bit code generator sequences used for this code are given in octal notation by

$$\mathbf{g}^{(1)} = 73353367672 \quad \mathbf{g}^{(2)} = 53353367672. \quad (6)$$

The code was chosen to be nonsystematic in order to give it a larger minimum free Hamming distance d_{free} , in this case $d_{\text{free}} = 21$, compared to the best systematic code of the same constraint length. This is true for convolutional codes in general, i.e., for a given constraint length, a measure of decoding complexity, more free distance, and thus better performance, can be achieved using a nonsystematic rather than a systematic code. The fact that the two generators differ in only one bit position gives the code the “quick-look” property, i.e., the capability of obtaining a reasonably accurate quick estimate of the information sequence from the noisy received sequence prior to actual decoding. This is an important capability in some situations that is always available with systematic codes, but not, in general, with nonsystematic codes. Requiring this capability does result in some reduction in free distance, however, and thus represents a compromise between choosing the best possible code and retaining the “quick-look” property. Nevertheless, the above code has had a long and distinguished career in space, having been used, in addition to the above two missions, on the Pioneer 12 Venus orbiter and the European Helios A and Helios B solar orbiter missions.

A sequential decoder using a modified version of the Fano tree-searching algorithm with 3-bit soft decisions (3-bit quantized demodulator outputs) was chosen for decoding. For lower speed operation, in the kilobit-per-second (kbps) range, decoding could be done in software. Faster hardware decoders were also developed for operation in the megabit-per-second (Mbps) range [12], [13]. This scheme had code rate $r = 1/2 = 0.5$ bit/signal and achieved a BER of 10^{-5} at an $E_b/N_0 = 2.7$ dB (see Fig. 2, “Pioneer”), thus achieving a 6.9-dB coding gain compared to uncoded BPSK, at the expense of a doubling in bandwidth requirements. This represented a significant improvement compared to the Mariner system and resulted in performance only 2.5 dB away from the BPSK capacity curve. An excellent discussion of the considerations that went into the design of these early deep-space coding systems is included in the paper by Massey [6].

Sequential decoding algorithms have a variable computation characteristic which results in large buffering requirements, and occasionally large decoding delays and/or incomplete decoding of the received sequence. In some situations, such as when almost error-free communication is required or when

retransmission is possible, this variable decoding delay property of sequential decoding can be an advantage. For example, when a long delay occurs in decoding, indicating a very noisy and therefore probably unreliable frame of data, the decoder can simply stop and erase the frame, not delivering anything to the user, or ask for a retransmission. A so-called “complete” decoder, on the other hand, would be forced to deliver a decoded estimate, which may very well be wrong in these cases, resulting in what has been termed a “fools rush in where angels fear to tread” phenomenon [11]. However, fixed delay is desirable in many situations, particularly when high-speed decoding is required. In addition, the performance of convolutional codes with sequential decoding is ultimately limited by the computational cutoff rate R_0 (the rate at which the average number of computations performed by a sequential decoder becomes unbounded), which requires SNR’s higher than capacity to achieve reliable communication at a given code rate [2], as shown in Fig. 2. For example, to achieve reliable communication at a code rate of $r = 0.5$ bit/signal using sequential decoding and BPSK modulation on the AWGN channel requires an $E_b/N_0 = 2.4$ dB, whereas the capacity bound only requires an $E_b/N_0 = 0.2$ dB. The E_b/N_0 at which the Pioneer code achieves a BER of 10^{-5} is only 0.3 dB away from the cutoff rate, and thus there is little to be gained with longer constraint length codes and sequential decoding at this code rate and BER.

These undesirable characteristics of sequential decoding and the possibility of higher decoding speeds led to the use of maximum-likelihood Viterbi decoding [14] in the next generation of deep-space communication systems. The Viterbi algorithm, like sequential decoding, is compatible with a variety of modulation and quantization schemes. Unlike sequential decoding, though, the Viterbi algorithm has a fixed number of computations per decoded branch and thus does not suffer from incomplete decoding and, ultimately, is not limited by a computational cutoff rate.

The Voyager 1 and 2 space missions were launched in 1977 to explore Jupiter and Saturn. They both used a $(2, 1, 7)$ nonsystematic convolutional code with generator polynomials

$$\begin{aligned} \mathbf{G}^{(1)}(D) &= 1 + D + D^3 + D^4 + D^6 \\ \mathbf{G}^{(2)}(D) &= 1 + D^3 + D^4 + D^5 + D^6 \end{aligned} \quad (7)$$

and $d_{\text{free}} = 10$. This code and a companion $(3, 1, 7)$ code with generators

$$\begin{aligned} \mathbf{G}^{(1)}(D) &= 1 + D + D^3 + D^4 + D^6 \\ \mathbf{G}^{(2)}(D) &= 1 + D^3 + D^4 + D^5 + D^6 \\ \mathbf{G}^{(3)}(D) &= 1 + D^2 + D^4 + D^5 + D^6 \end{aligned} \quad (8)$$

and $d_{\text{free}} = 15$, were both adopted as NASA/ESA Planetary Standard Codes by the Consultative Committee on Space Data Systems (CCSDS). The $(2, 1, 7)$ code has also been employed in numerous other applications, including satellite communication and cellular telephony, and has become a *de facto* industry standard [15].

The above codes were decoded using a 3-bit soft-decision maximum-likelihood Viterbi decoder. Since the complexity

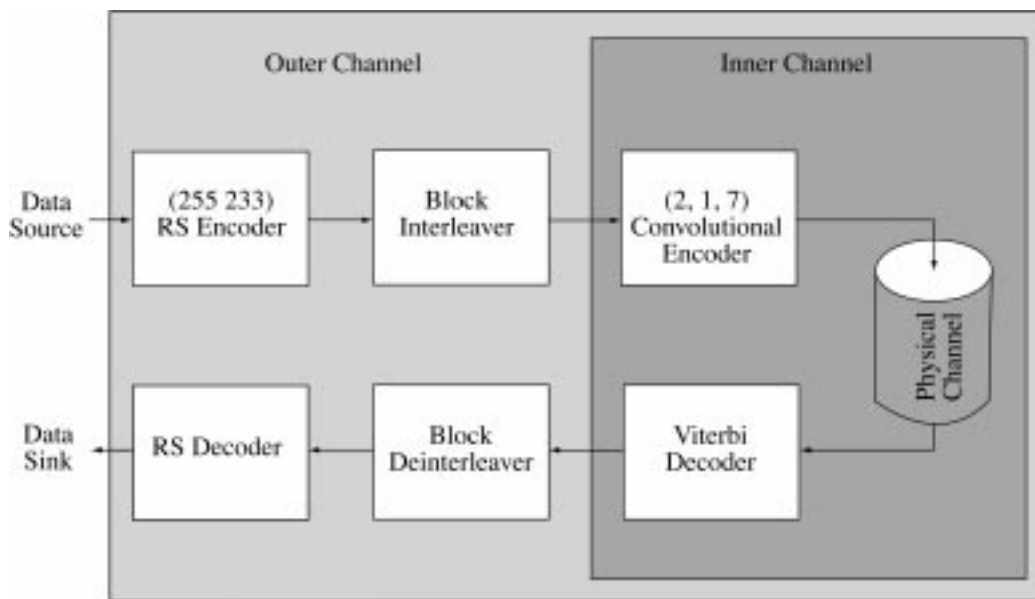


Fig. 3. The CCSDS concatenation standard.

of Viterbi decoding grows exponentially with code constraint length, it was necessary to choose these short constraint length codes rather than the long constraint length Pioneer codes used with sequential decoding. The $K = 7$ codes chosen have a decoding trellis containing 64 states, considered reasonable in terms of implementation complexity. The performance of these codes is plotted in Fig. 2 with the label "Planetary Standard". The $(2, 1, 7)$ code requires an $E_b/N_0 = 4.5$ dB to operate at a BER of 10^{-5} . Though this code results in a 5.1-dB power advantage compared to uncoded transmission, its performance is 1.8 dB worse than the Pioneer system, due to the short constraint length used. However, its decoder implementation complexity is simpler than a sequential decoder, it does not suffer the long buffering delays characteristic of sequential decoding, and, because of its regular trellis structure, it is adaptable to parallel implementation, resulting in decoding speeds in the 100's of Mbps [16].

The Planetary Standard also played a major role in military satellite communications well into the 1980's (as incorporated into the Satellite Data Link Standard (SDLS)). In general, convolutional encoding with Viterbi decoding will continue to be used in earth-orbiting satellite communication systems well into the next century. The Globalstar and Iridium systems use $K = 9$, rate $1/2$ and $K = 7$, rate $3/4$ convolutional codes, respectively. The rationale for the differing constraint lengths and rates lies with the nominal lengths (and consequent space loss) of the satellite-to-ground links for the two systems. Globalstar satellites operate at altitudes of approximately 1400 km, while Iridium operates at half that height [17].

Coding gain beyond that provided by the Planetary Standard can be achieved using code concatenation. Concatenation is a scheme first introduced by Forney [18] in which two codes, an "inner code" and an "outer code," are used in cascade. The inner code should be designed to produce a moderate BER (typically on the order of 10^{-3} to 10^{-4}) with modest complexity. The outer code can be more complex and should

be designed to correct almost all the residual errors from the inner decoder, resulting in nearly error-free performance (BER's, say, on the order of 10^{-10}). The most common arrangement is a combination of a short constraint length inner convolutional code with soft-decision Viterbi decoding and a powerful nonbinary Reed-Solomon (RS) outer code. This combination was eventually accepted in 1987 as the CCSDS Telemetry Standard [19].

In fact, though, several other concatenation schemes had been tried earlier. For example, on the 1971 Mariner mission, a $(6, 4)$ RS outer code with symbols drawn from $GF(2^6)$ was used in conjunction with the $(32, 6)$ RM code as an inner code, and on the two 1977 Voyager missions, a $(24, 12)$ extended Golay outer code was used together with the $(2, 1, 7)$ Planetary Standard code as an inner code [20]. In both cases, the data to be transmitted consisted mostly of uncompressed image information, along with small amounts of sensitive scientific information. The outer codes were used only to give added protection to the scientific information, so that the overall coding rate was not reduced much below the inner-code rates. In the case of the Mariner system, each 6-bit symbol from the outer code is encoded into one 32-bit codeword in the inner code. This "matching" between the outer code symbol size and the information block size of the inner code means that each block error from the inner decoder causes only one symbol error for the outer decoder, a desirable property for a concatenation scheme consisting of two block codes. Finally, although both inner decoders made use of soft-decision inputs from the channel, the outer decoders were designed to work directly with the "hard decisions" made by the inner decoders. Outer decoders which also make use of soft-decision inputs will be considered later in this section.

The CCSDS Standard concatenation scheme consists of the $(2, 1, 7)$ Planetary Standard inner code along with a $(255, 223)$ RS outer code, as shown in Fig. 3. (Note that the CCSDS Standard assumes that all the data is protected by

both codes, inner and outer, although it is clearly possible to protect some data using only the inner code or to send some data without any protection at all.) The RS code consists of 8-bit symbols chosen from the finite field $GF(2^8)$ based on the primitive polynomial

$$\mathbf{p}(x) = x^8 + x^7 + x^2 + x + 1. \quad (9)$$

The generator polynomial (in cyclic code form) is given by

$$\mathbf{g}(x) = \prod_{j=1}^{143} (x - \alpha^{11j}) \quad (10)$$

where α is a root of $\mathbf{p}(x)$. From (10), we see that $\mathbf{g}(x)$ has 32 first-order roots, giving it degree 32, and thus the code contains 32 redundant symbols. Since RS codes are maximum-distance separable (MDS), their minimum distance is always one more than the number of redundant symbols. Hence this code has $d_{\min} = 33$ and can correct any combination of 16 or fewer symbol errors within a block of 255 symbols (2040 bits). Hard-decision decoding of the outer code is performed using the Berlekamp–Massey algorithm [21], [22]. Finally, in order to break up possibly long bursts of errors from the inner decoder into separate blocks for the outer decoder, thus making them easier to decode, a symbol interleaver is inserted between the inner and outer codes. Interleaver depths of between two and eight outer-code blocks are typical [23].

With the CCSDS Standard concatenation scheme, the E_b/N_0 needed to achieve a BER of 10^{-5} is reduced by a full 2.0 dB compared to using the Planetary Standard code alone, with only a slight reduction in code rate (from $r = 0.5$ to $r = 0.44$). The performance of the (2, 1, 7) code in a concatenated system with the (255, 223) RS outer code is shown in Fig. 2 with the label “Voyager.” The concatenated Voyager system operates in the same E_b/N_0 region as the Pioneer system, gaining about 0.2 dB with a 12.5% loss in rate. Thus short constraint length convolutional codes with Viterbi decoding in concatenated systems can be considered as alternatives to long constraint length codes with sequential decoding.

Variations on the concatenated CCSDS theme continue to be used at the end of the century in near-earth applications. For example, in 1998 the DirecTV satellite system was using a concatenated convolutional-Viterbi/Reed–Solomon system to bring television signals to three million subscribers.

Recently, technological advances have made it practical to build maximum-likelihood Viterbi decoders for larger constraint length convolutional codes. The culmination of this effort was the Big Viterbi Decoder (BVD) designed by Collins to decode a (4, 1, 15) code. The BVD was constructed at JPL for use on the Galileo mission to Jupiter [24]. The decoder trellis has $2^{14} = 16384$ states, making internal decoder communication a formidable task, but decoding speeds up to 1 Mbps were still achieved. The code has octal generators

$$\mathbf{g}^{(1)} = 46321 \quad \mathbf{g}^{(2)} = 51271 \quad \mathbf{g}^{(3)} = 63667 \quad \mathbf{g}^{(4)} = 70535 \quad (11)$$

and $d_{\text{free}} = 35$, clearly a very powerful code. It achieves a BER of 10^{-5} at an $E_b/N_0 = 1.7$ dB, only 2.6 dB away from

the BPSK capacity curve. (See Fig. 2, “BVD.”) Compared to the (2, 1, 7) Planetary Standard code, it gains a full 2.8 dB. In a concatenated system with the (255, 223) RS outer code, the (4, 1, 15) code requires an E_b/N_0 of just 0.9 dB to achieve a BER of 10^{-5} , is within 2.0 dB of capacity, and is 1.6 dB more power efficient than the Voyager system. (See Fig. 2, “Galileo.”) Although the above rate 1/4 systems are 50% less bandwidth efficient than their rate 1/2 counterparts, it should be recalled that bandwidth is plentiful in deep space, and thus it is common to sacrifice spectral efficiency for added power efficiency. In fact, an even less spectrally efficient (6, 1, 15) code is currently scheduled to be flown aboard the Cassini mission to Saturn [25]. However, further bandwidth expansion may be difficult to achieve due to the fact that additional redundancy may reduce the energy per transmitted symbol below the level needed for reliable tracking by the phase-locked loops in the coherent demodulator.

As a further improvement on the CCSDS Concatenation Standard, errors-and-erasures decoding, a suboptimum form of soft-decision decoding, can be used to provide some performance improvement if erased symbols are available from the inner decoder. One method of providing erased symbols is based on two facts mentioned above: 1) a frame of several RS code blocks is interleaved prior to encoding by the inner code, and 2) decoding errors from the inner decoder are typically bursty, resulting in strings of consecutive error symbols. Although long strings of error symbols will usually cause problems for an RS decoder, after de-interleaving they are more spread out, making them easier to decode. In addition, once a symbol error has been corrected by the RS decoder, symbols in the corresponding positions of the other codewords in the same frame can be flagged as erasures, thus making them easier to decode. This technique is known as “error forecasting” and has been discussed in a paper by Paaske [26].

Another method of improving the CCSDS Concatenation Standard makes use of iterative decoding. In one approach, the RS codes in a given frame are assigned different rates, some higher than the (255, 223) code and some lower, such that the average rate is unchanged. After an initial inner decoding of one frame, the most powerful (lowest rate) outer code is decoded, and then its decoded information bits (correct with very high probability) are fed back and treated as known information bits (side information) by the inner decoder in a second iteration of decoding. This procedure can be repeated until the entire frame is decoded, with each iteration using the lowest rate outer code not yet decoded. The use of known information bits by the inner decoder has been termed “state pinning,” and the technique is discussed in a paper by Collins and Hizlan [27].

A more general approach to iterative decoding of concatenated codes was proposed by Hagenauer and Hoehner with the introduction of the Soft-Output Viterbi Algorithm (SOVA) [28]. In the SOVA, reliability information about each decoded bit is appended to the output of a Viterbi decoder. An outer decoder which accepts soft inputs can then use this reliability information to improve its performance. If the outer decoder also provides reliability information at its output, iterative decoding can proceed between the

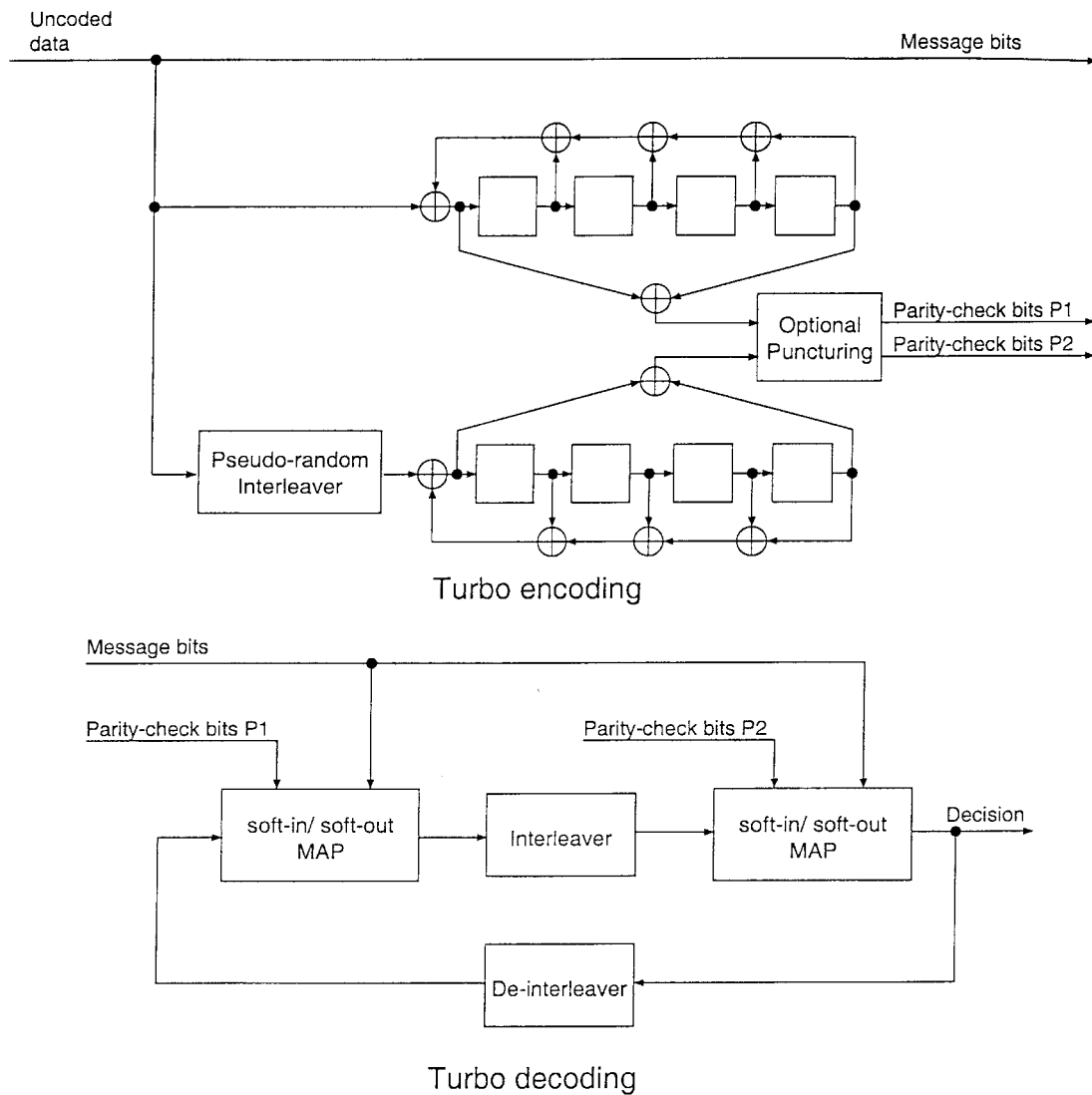


Fig. 4. The “turbo” encoding/decoding system.

inner and outer decoders. In general, such iterative decoding techniques for concatenated systems can result in additional coding gains of up to about 1.0 dB. In the CCSDS system, however, the outer RS decoder cannot make full use of such reliability information. Nevertheless, several combinations of error forecasting, state pinning, and iterative decoding have been applied to the CCSDS system by various researchers, resulting in an additional coding gain of about 0.5 dB [20], [23].

As our final topic in this section, we discuss a significant new discovery called “turbo codes,” which is currently being considered as a software upgrade for the Cassini mission. Turbo codes, which are also known as parallel concatenated convolutional codes, were first introduced in a paper by Berrou, Glavieux, and Thitimajshima [29]. Turbo codes combine a convolutional code along with a pseudorandom interleaver and maximum *a posteriori* probability (MAP) iterative decoding to achieve performance very close to the Shannon limit. A block diagram of the turbo encoding/decoding system is shown in Fig. 4. The encoder employs a simple (2, 1, 5) code in systematic feedback form, using two copies of the

parity generator separated by a pseudorandom interleaver. The generator matrix is given by

$$\mathbf{G}(D) = \left[1 \quad \frac{1 + D^4}{1 + D + D^2 + D^3 + D^4} \right]. \quad (12)$$

(The code is the same as that generated by a conventional non-systematic feedforward encoder with generator polynomials $\mathbf{G}^{(1)}(D) = 1 + D + D^2 + D^3 + D^4$ and $\mathbf{G}^{(2)}(D) = 1 + D^4$, but it is important that it be encoded in systematic feedback form because of the way the interleaver combines the two parity sequences.) The encoder output consists of the information sequence and two parity sequences, thus representing a code rate of 1/3. Alternately puncturing (deleting) bits from the two parity sequences produces a code rate of 1/2, and other code rates can be achieved by using additional parity generators and/or different puncturing patterns. The pseudorandom interleaver re-orders the information sequence before encoding by the second parity generator, thus producing two different parity sequences. In essence, the interleaver has the effect of matching “bad” (low-weight) parity sequences with “good” (higher weight) parity sequences in almost all cases, thus

generating a code with very few low-weight codewords. For large enough information sequence blocklengths, performance very close to the Shannon limit can be achieved at moderate BER's, even though the free distance of turbo codes is not large. In fact, for most interleavers, the free distance of the above code is only $d_{\text{free}} = 6$, even for very long blocklengths. The excellent performance at moderate BER's is due rather to a drastic reduction in the number of nearest neighbor codewords compared to a conventional convolutional code. In a paper by Benedetto and Montorsi [30], it was shown that the number of nearest neighbors is reduced by a factor of N , where N is the blocklength. This factor is referred to as the "interleaver gain."

The other important feature of turbo codes is the iterative decoder, which uses a soft-in/soft-out MAP decoding algorithm first applied to convolutional codes by Bahl, Cocke, Jelinek, and Raviv [31]. This algorithm is more complex than the Viterbi algorithm by about a factor of three, and for conventional convolutional codes it offers little performance advantage over Viterbi decoding. However, in turbo decoding, the fact that it gives the maximum MAP estimate of each individual information bit is crucial in allowing the iterative decoding procedure to converge at very low SNR's. Although the SOVA can also be used to decode turbo codes, significant improvement can be obtained with MAP decoding [32].

At almost any bandwidth efficiency, performance less than 1.0 dB away from capacity is achievable with short constraint length turbo codes, very long blocklengths, and 10–20 iterations of decoding. In Fig. 2, the point marked "Turbo Code" shows performance at a BER of 10^{-5} for rate $r = 1/2$ and information blocklength $N = 2^{16} = 65536$, with 18 iterations of decoding. This is a full 3.8 dB better than the Planetary Standard (2, 1, 7) code, with roughly the same decoding complexity, and is also 1.0 dB better than the very complex BVD code, which operates at 50% less spectral efficiency! Using the same rate of $1/4$, the turbo code outperforms the BVD code by about 1.9 dB. The major disadvantages of a turbo code are its long decoding delay, due to the large blocklengths and iterative decoding, and its weaker performance at lower BER's, due to its low free distance. The long delays are not a major problem except in real-time applications such as voice transmission, and performance at lower BER's can be enhanced by using serial concatenation [33], so turbo codes seem to be ideally suited for use on many future deep-space missions.

A comprehensive survey of the application of coding to deep-space communication as the decade of the 1960's drew to a close was given in a paper by Forney [34]. For a more recent review of the subject, the article by Wicker [25] is an excellent source.

III. APPLICATIONS TO DATA TRANSMISSION

As mentioned previously in this paper, the focus of much of the early work in coding theory was on reducing power requirements at low spectral efficiencies. This was due, in part, to the fact that no practical coding schemes existed that could provide meaningful power gains at higher spectral efficiencies.

This changed dramatically with Ungerboeck's discovery of trellis-coded modulation (TCM) [35].

In his 1982 paper, Ungerboeck constructed trellis codes for amplitude modulation (AM), phase-shift-keyed (PSK), and quadrature amplitude modulation (QAM) modulation schemes. As an example, consider the simple four-state, 8-PSK code shown in Fig. 5 along with its trellis diagram. The encoder has two input bits, x^2 and x^1 . The input bit x^2 is differentially encoded to produce the encoded bit y^2 . (Differential encoding of this bit is needed to take advantage of the "rotational invariance" property of the code so that the correct information sequence can be recovered after decoding in the event the receiver demodulator locks onto the wrong phase. This is an important practical consideration which will be discussed further later in this section.) The input bit x^1 is encoded by a (2, 1, 3) systematic feedback convolutional encoder with generator matrix

$$\mathbf{G}(D) = \left[1 \quad \frac{D}{1+D^2} \right] \quad (13)$$

producing the two encoded bits $y^1 = x^1$ (an information bit) and y^2 (a parity bit). (It should be noted here that the reason for using systematic feedback encoders in TCM has nothing to do with the reason for using them in turbo coding. In TCM, nonsystematic feedforward encoders can be used with similar results, but systematic feedback encoders are preferred because they provide a convenient canonical representation for the high-rate codes, typically $r = k/(k+1)$, commonly used in TCM and because the noisy received information bits can be directly accessed prior to decoding.) The signal mapper (modulator) then generates an 8-PSK signal according to the mapping shown in Fig. 5. Ignoring the differential encoding, at each time unit the mapper input consists of two information bits, y^2 and y^1 , and one parity bit, y^0 , and the mapper output consists of one 8-PSK signal. Thus the information rate, or spectral efficiency, of this scheme is $\eta = 2$ bits/signal, the same as an uncoded system using QPSK modulation. (The fact that only one information bit, y^1 , enters the convolutional encoder and the other information bit, y^2 , remains "uncoded" is a feature of many, but not all, TCM schemes. The best code design for a given situation may or may not contain uncoded information bits, depending on the signal constellation, code complexity, and spectral efficiency.) Decoding is performed by a conventional soft-decision Viterbi decoder operating directly on the noisy received 8-PSK signals. In this example, the decoding trellis has four states, as shown in Fig. 5. Because the code has one uncoded information bit, the trellis contains so-called "parallel transitions," i.e., parallel paths of length one branch each connecting pairs of states at different time units. Parallel transitions in the trellis diagram are characteristic of all TCM schemes that contain uncoded information bits.

The first important insight due to Ungerboeck was his realization that for the nonbinary signal constellations used in TCM, minimum free squared distance d_{free}^2 in Euclidean space, rather than minimum free Hamming distance d_{free} , is the primary parameter that determines code performance. His technique of "mapping by set partitioning," as opposed to the conventional Gray mapping used in uncoded modulation,

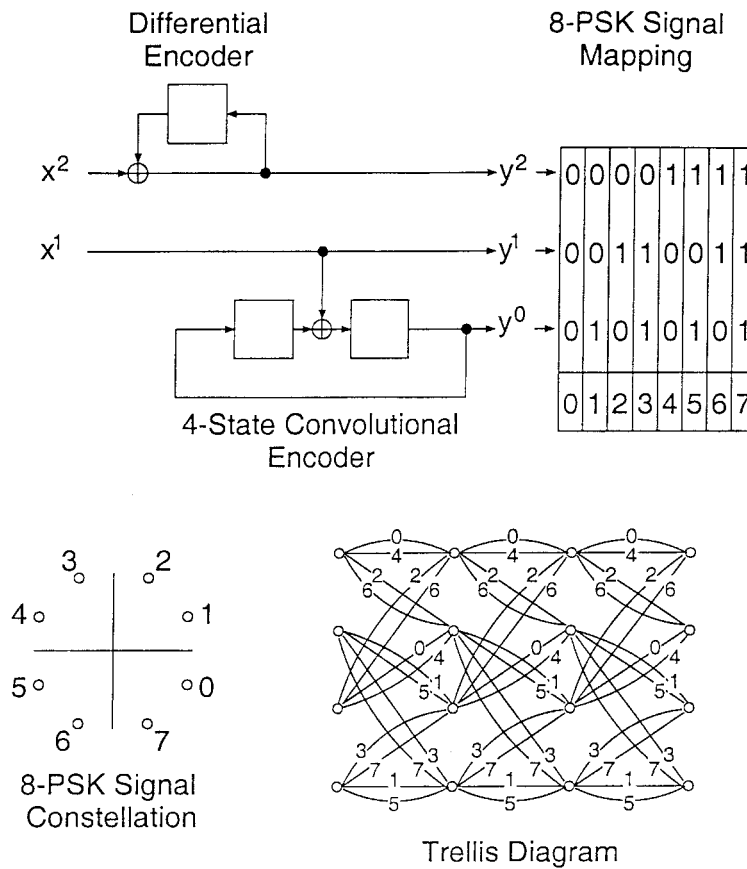


Fig. 5. The four-state 8-PSK Ungerboeck code.

turned out to be the key to achieving performance gains with TCM compared to uncoded modulation at high spectral efficiencies. One of the important goals of mapping by set partitioning is to insure that parallel transitions in the trellis are mapped into signals far apart in signal space, thus minimizing the probability of short, one-branch, error events due to uncoded bits. For the four-state, 8-PSK code considered above, $d_{\text{free}}^2 = 4$, whereas for uncoded QPSK, $d_{\text{free}}^2 = 2$, assuming unit energy signals in both cases. Hence this simple TCM scheme has a factor of 2, or 3 dB, asymptotic, i.e., high SNR, coding gain compared to uncoded QPSK. (The asymptotic coding gain of any code is always somewhat larger than its “real” coding gain at moderate BER’s, due to the effect of nearest neighbor codewords. In this case, the coding gain at a 10^{-5} BER is 2.4 dB, as shown in Fig. 1 with the notation “(2, 1, 3) 8-PSK.”) Also, unlike the case with binary modulation, where coding gains are achieved only at the expense of bandwidth expansion, the TCM scheme has the same spectral efficiency, viz., 2.0 bits/signal, as uncoded QPSK. Thus the 3-dB asymptotic coding gain is achieved without bandwidth expansion! In the case of TCM, it is expanding the size of the signal constellation, rather than expanding the bandwidth, that results in the possibility of coding gain. This fundamental realization by Ungerboeck has led to an entire new set of application areas for coding in the range of high spectral efficiencies.

Ungerboeck was able to construct a number of interesting codes as a result of his original work [36], [37]. For

example, his 64-state, (2, 1, 7), 4-ary AM code requires an $E_b/N_0 = 6.2$ dB to achieve a BER of 10^{-5} with soft-decision Viterbi decoding. This code has the same spectral efficiency as uncoded BPSK and requires 3.4 dB less power to achieve the same BER! Note, however, that a (2, 1, 7) convolutional code can also be used with conventional Gray mapped QPSK modulation to achieve a BER of 10^{-5} with an $E_b/N_0 = 4.5$ dB, 1.7 dB less than the AM code, thus illustrating the advantage of using quadrature modulation when possible. (See Fig. 1, “(2, 1, 7) QPSK.”) As another example, Ungerboeck’s 64-state, (3, 2, 7), 8-PSK code has a spectral efficiency of $\eta = 2.0$ bits/signal and requires an E_b/N_0 of only 6.0 dB to reach a BER of 10^{-5} with soft-decision Viterbi decoding. (See Fig. 1, “(3, 2, 7) 8-PSK.”) Thus the four-state and 64-state, 8-PSK Ungerboeck codes require 2.4 and 3.6 dB less power, respectively, than uncoded QPSK to achieve the same BER with the same spectral efficiency. Finally, the 64-state code is 3.3 dB away from the 8-PSK capacity curve, which means that more than half the potential gain of rate $r = 2/3$, 8-PSK TCM can be achieved with relatively modest complexity.

For spectral efficiencies of $\eta = 3.0$ bits/signal or greater, two-dimensional (2D) or multidimensional (MD) QAM signal sets offer better performance than PSK signal sets. (MD signal sets are formed by mapping encoded bits into more than one constituent 2D signal point at a time.) QAM signal sets more efficiently pack the 2D signal space and thus can achieve the same minimum distance between signal points,

which effectively limits the performance of a trellis code, with less average energy per signal than a comparable PSK signal set. This performance advantage of QAM signal sets has led modem designers to choose QAM, and QAM-based trellis codes, for use in high-speed data transmission.

The performance of Ungerboeck's codes quickly dispelled the belief that power reduction is only attainable with a corresponding decrease in spectral efficiency, as is the case when we limit our choice of modulation to BPSK. This was a welcome result for modem designers, who had been frustrated in their attempts to go beyond data rates of 9600 bps [38]. Since the International Telecommunications Union's ITU-T V.29 modem standard was adopted in 1976, little progress was made in increasing the speed and quality of data transmission over voice-grade telephone lines until the appearance of the V.32 and V.33 standards in 1986, both of which included a TCM encoder/modulator. The V.29 standard used uncoded 16-QAM modulation and a 2400-symbol/s signaling rate to achieve a spectral efficiency of $\eta = 4.0$ bits/signal and a transmission speed of 9600 bps in a half-duplex (one-way) mode. Due to the bandwidth constraints of the channel, signaling rates higher than 2400 symbols/s were not considered feasible. Thus the only avenue to increased data rates was to expand the size of the signal constellation. However, due to the SNR constraints of the channel, this meant that signals had to be packed closer together, resulting in degraded performance. So a clear need developed for a scheme which would allow constellation expansion at the same signaling rate, thus achieving higher data rates, and yet provide a coding gain to at least recover the noise margin lost by the closer packing of signals. TCM proved to be just such a scheme and, combined with some sophisticated signal-processing techniques, has resulted in a series of improvements which have pushed modem speeds up to 33 600 bps full-duplex.

In the remainder of this section, we will limit our discussion to two examples: the eight-state, (3, 2, 4), nonlinear 2D code developed by Wei [39] and adopted by ITU-T for the V.32 and V.33 standards and the 16-state, (3, 2, 5), linear four-dimensional (4D) code also developed by Wei [40] and adopted by ITU-T as one of the codes for the V.34 standard. The V.32 standard uses two uncoded bits and a 32-CROSS signal constellation, for a spectral efficiency of $\eta = 4.0$ bits/signal and a 9600 bps data rate, and achieves about 3.5 dB better performance (due to trellis coding) and full-duplex operation (due to echo cancellation) compared to the uncoded V.29 standard operating at the same data rate. In the V.33 standard, four uncoded bits and a 128-CROSS constellation are used to achieve $\eta = 6.0$ bits/signal and a 14 400-bps data rate. The V.34 standard, considered the ultimate modem standard for the classic linear Gaussian model of the telephone channel [41], uses an MD mapping, a constituent 2D signal constellation containing a maximum of 1664 points, and as many as eight uncoded bits (ten information bits total) per symbol to achieve a spectral efficiency up to $\eta = 10.0$ bits/signal and data rates as high as 33 600 bps. Advanced line probing, adaptive equalization, and precoding techniques are used to increase the signaling rates to as high as 3429 symbols/s, compared to

the previous standard of 2400 symbols/s. (In V.34, the data rates are not necessarily integer multiples of the symbol rates, since the signal mapping technique, called shell mapping [42], allows the mapping of fractional bits per symbol.) The V.34 standard also includes two other codes which offer slightly better performance at a cost of increased complexity [41]. One is a 32-state, (4, 3, 6), linear 4D code developed by Williams [43] that gains 0.3 dB compared to the 16-state Wei code, but is four times more complex. The other is a variation of a 64-state, (5, 4, 7), nonlinear 4D code also developed by Wei [40] that gains an additional 0.15 dB over the 16-state code, but is 16 times as complex. In both these cases, more bits are encoded by the convolutional code, so fewer uncoded bits are needed to achieve a particular data rate.

A block diagram of the eight-state, nonlinear 2D code developed by Wei [39] is shown in Fig. 6, along with a sketch of the 32-CROSS constellation adopted for the ITU-T V.32 standard. Note that the encoder has four input information bits, x^1, x^2, x^3 , and x^4 . $x^3 = y^3$ and $x^4 = y^4$ are uncoded and directly enter the 32-CROSS signal mapper (modulator). x^1 and x^2 are first differentially encoded and then enter the (3, 2, 4) systematic feedback nonlinear convolutional encoder, producing three output bits: y^1 and y^2 (information bits) as well as y^0 (a parity bit). The five encoded bits y^0, y^1, y^2, y^3 , and y^4 then enter the modulator and are mapped into one of the 32-CROSS signals, as shown in Fig. 6. Since one 32-CROSS signal is transmitted for every four information bits entering the encoder, the spectral efficiency of the code is $\eta = 4.0$ bits/signal. (The V.33 standard uses the same code along with four uncoded information bits and a 128-CROSS constellation to achieve a spectral efficiency of $\eta = 6.0$ bits/signal.) As noted in the above example, soft decision Viterbi decoding, using an eight-state trellis with four-fold (16-fold in the V.33 case) parallel transitions, is performed based on the noisy received symbols at the demodulator output. Since the encoder contains two AND gates, it is nonlinear and cannot be described using the usual generator sequence notation. The nonlinear encoder was needed to make the code invariant to 90° phase rotations, i.e., a rotation of the signal space by any multiple of 90° still results in a valid code sequence. For a code with this property, as long as the bits affected by the phase rotation are differentially encoded, the correct sequence can still be decoded if the demodulator locks onto the wrong phase by any multiple of 90° . For 2D signal constellations, it is impossible to achieve 90° invariance with linear codes (the best that can be done is 180° invariance), and hence nonlinear codes are needed for full rotational invariance. This was a crucial insight made by Wei [39] in the design of this code.

The eight-state, (3, 2, 4), nonlinear 2D Wei code used in the V.32 and V.33 standards has free distance $d_{\text{free}}^2 = 5$, number of nearest neighbors $N_{\text{free}} = 16$, and achieves a coding gain of 3.6 dB at a BER of 10^{-5} compared to uncoded 16-QAM ($\eta = 4.0$) and 64-QAM ($\eta = 6.0$) modulation, respectively. As in the other examples, this coding gain is achieved without bandwidth expansion. These points are shown in Fig. 1 with the designations "V.32" and "V.33."

A block diagram of the 16-state, (3, 2, 5), linear 4D code developed by Wei [40] is shown in Fig. 7, along with a

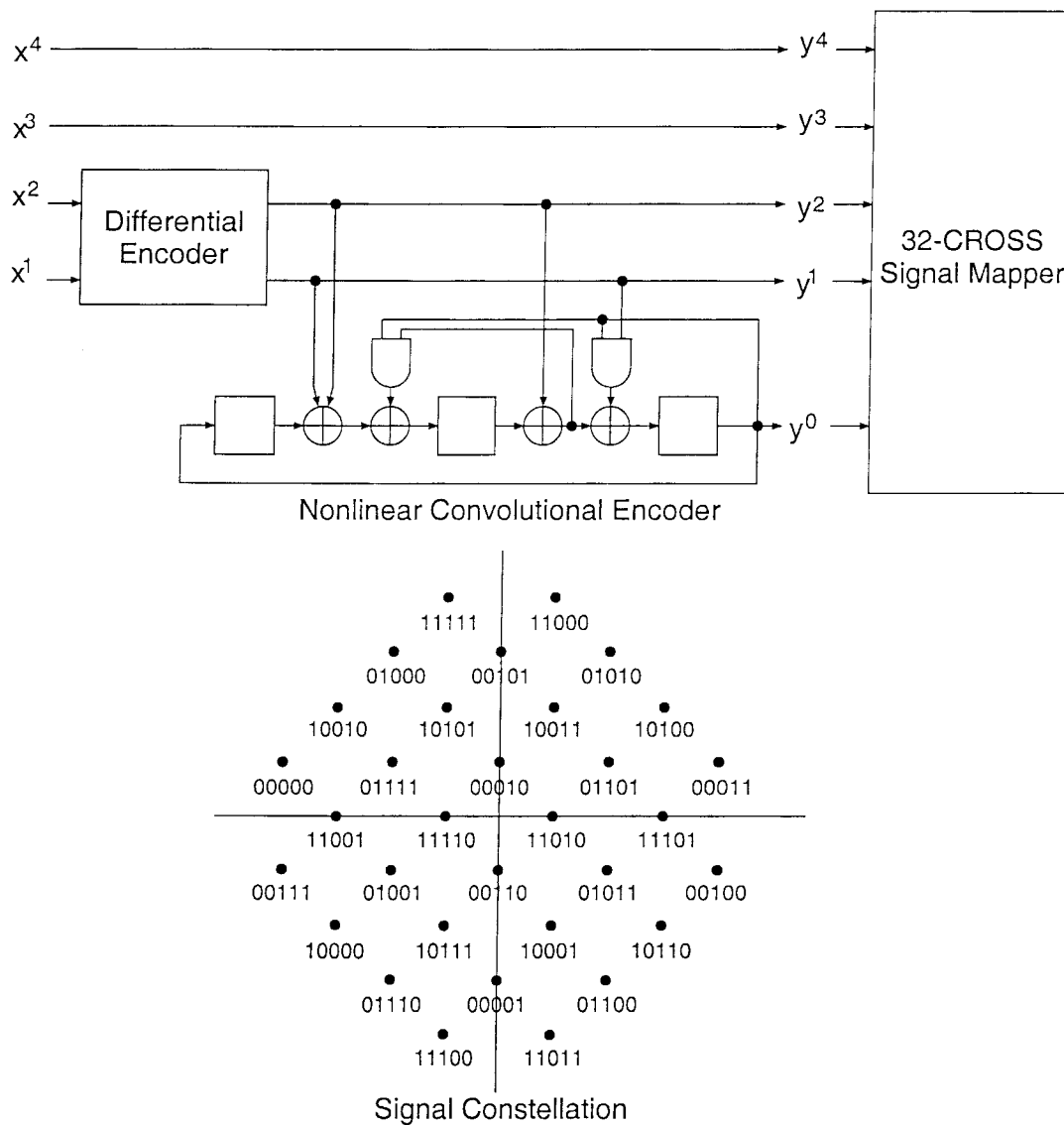


Fig. 6. The eight-state V.32 code.

sketch of a 224-point constellation which, along with its mirror inverse, forms a constituent 448-point 2D signal constellation consistent with the ITU-T V.34 standard. The TCM portion of the encoder has three input information bits: two coded (x^1 and x^2) and one uncoded (x^3). In addition, bits x^2 and x^3 are differentially encoded, since they are affected by phase rotations. Full 90° rotational invariance can be achieved with a linear code in this case since the signal constellation is 4D. (In general, it is possible to achieve full rotational invariance with linear codes using MD signal constellations, at least for code rates $r = k/(k + 1)$ with $k < 4$.) The $(3, 2, 5)$ systematic feedback linear convolutional encoder shown in Fig. 7 (a simplified form) is equivalent to an encoder with generator matrix

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & \frac{D^3}{1+D^4} \\ 0 & 1 & \frac{1+D}{1+D^4} \end{bmatrix} \quad (14)$$

and produces three output bits: y^1 and y^2 (information bits) as well as y^0 (a parity bit). Along with the differentially

encoded bit y^3 , they enter the 4D signal mapper (modulator). The constituent 2D constellation is divided into four subsets, and thus there are 16 subsets in the 4D constellation. The four encoded bits y^0, y^1, y^2 , and y^3 are mapped into one of the 16 subsets in the 4D constellation. In the most basic version of the standard, the additional uncoded information bits needed to achieve the desired spectral efficiency are then used to select the particular 4D signal point to be transmitted. For example, if a spectral efficiency of $\eta = 7.0$ bits/signal is desired using a constituent 192-point 2D signal constellation, an additional 11 uncoded information bits are needed in each 4D signaling interval. In this case, the ratio of the constellation size to the size of the uncoded constellation which would support the same spectral efficiency, i.e., the constellation expansion ratio (CER), is $\text{CER} = 192/128 = 1.5$, or 50% less than that needed with 2D TCM. By choosing the constituent 2D constellation subsets and the mapping such that lower energy signal points (the “inner points” in the constellation) are used more often than higher energy points (the “outer

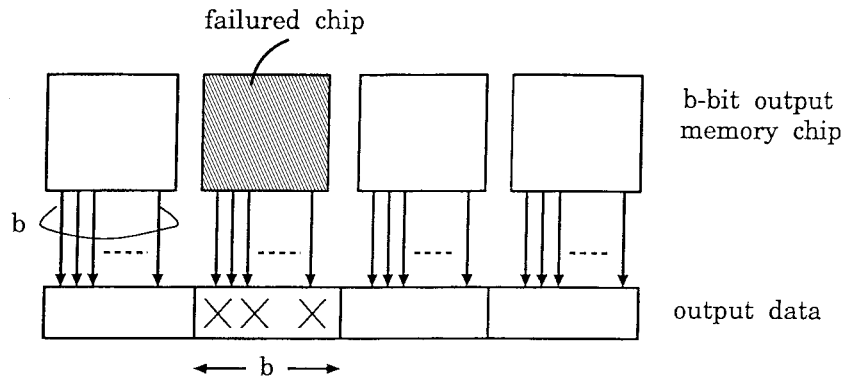


Fig. 8. The b -bit-per-chip semiconductor memory organization.

points” in the constellation) a technique called constellation shaping [44], an additional “shaping gain” is achieved. This concept can be extended by further expanding the basic 2D signal constellation beyond 192 points in conjunction with the shell mapping approach. In this approach, the signal selection (mapping) is done over several symbol intervals so that enough dimensions are included to produce an approximately spherical constellation, known to be optimal in high-dimensional spaces. In the V.34 standard, shell mapping is applied to a 16-dimensional constellation, i.e., eight 2D symbol intervals, to achieve up to 0.8 dB of shaping gain. For the constituent 448-point 2D signal constellation shown in Fig. 7, a total of 20 information bits over eight 2D symbols are devoted to shell mapping, and four information bits per symbol are left uncoded. This arrangement gives a total spectral efficiency of

$$\begin{aligned} \eta &= 3/2 \text{ (coding)} + 20/8 \text{ (shaping)} + 4 \text{ (uncoded)} \\ &= 8.0 \text{ bits/signal.} \end{aligned} \quad (15)$$

The size of the constituent 2D signal set can vary depending on the desired shaping gain and CER. Usually, an additional CER due to shaping of about 25% is enough to approach 0.8-dB shaping gains. In this example, the constellation size of 448 yields a $\text{CER} = 448/256 = 1.75$, for a total of 75% constellation expansion, 50% due to coding and 25% due to shaping. Shaping was included in the standard because it was ultimately a simpler way of picking up an additional 0.8 dB than by using a more complex code. In theory, as much as 1.5 dB of shaping gain is available [5], but the methods needed to achieve the extra gain are quite complex.

The 16-state, $(3, 2, 5)$, linear 4D Wei code used in the V.34 standard has free distance $d_{\text{free}}^2 = 4$, number of nearest neighbors $N_{\text{free}} = 12$ per 2D signal, and achieves a coding gain of 4.2 dB at a BER of 10^{-5} compared to uncoded 256-QAM ($\eta = 8.0$) modulation. (Note that the smaller free distance here compared to the eight-state V.32 code does not translate into less coding gain, since the uncoded systems used to compute the coding gain are different). This point is shown in Fig. 1 with the designation “V.34/16.” The slightly improved performance obtained by using 32- and 64-state codes is also indicated in Fig. 1 with the designations “V.34/32” and “V.34/64,” respectively.

A good summary of the state-of-the-art in modem design circa 1990 was given in the review article by Forney [38]. A

more thorough discussion of the considerations that went into the design of the V.34 modem can be found in [41].

IV. DATA STORAGE APPLICATIONS

In this section, we cover two applications of coding to data storage, high-speed computer memories and magnetic/optical discs. The use of RS codes in compact discs represents the most widely used and best known application of error-control coding over the 50-year time span covered by this review.

A. Error-Correcting Codes for High-Speed Computer Memories

Error-correcting codes are widely used to enhance the reliability of computer memories. In these storage applications, the codes have unique features not seen in most communication applications. This is due to the following conditions required for error control in computer memories.

- 1) Encoding and decoding must be very fast to maintain high throughput rates. Therefore, parallel encoding and decoding must be implemented with combinatorial logic instead of the linear feedback shift-register logic usually used in other applications.
- 2) Unique types of errors, such as byte errors, are caused by the organization of semiconductor memory systems. As memory systems become large, the b -bit-per-chip organization shown in Fig. 8 tends to be adopted. Then a chip failure causes the word read out to have errors in a b -bit block, called a b -bit byte or simply a byte.
- 3) Redundancy must be small, because the cost per bit of high-speed semiconductor memories is high compared with magnetic memories and other low-speed memories.

The best-known class of codes for semiconductor memories is the SEC/DED (single-error-correcting/double-error-detecting) codes. It can be viewed as a class of shortened versions of extended Hamming codes. These codes have minimum distance $d_{\text{min}} = 4$ and are capable of correcting single errors and detecting double errors. Usually, however, they can detect more than just double errors. Since the additional error-detection capability depends on the structure of the parity-check matrix, many proposals for enhanced error-detection properties have been presented.

Among the many proposed schemes, the SEC/DED/SbED (single-error-correcting/double-error-detecting/single b -bit byte error-detecting) codes have been put into practical use. These codes can correct a single bit error and detect any errors in a single b -bit byte as well as any two bit errors. Such detection capability is useful for the memory organization of Fig. 8, i.e., a b -bit per chip organization, where a chip failure causes a byte error.

As an example, let us consider the case of $b = 4$. An SEC/DED/S4ED code is constructed as follows. Let $p = 2^{m-1}$ and let $\{\mathbf{h}_i | i = 1, \dots, p\}$ be the set of all binary m -dimensional column vectors of even weight if m is odd and of odd weight if m is even. For $i, j = 1, 2, \dots, p$, the following $2m \times 4$ matrices \mathbf{H}_{ij} are defined as

$$\mathbf{H}_{ij} = \begin{bmatrix} \mathbf{h}_i & \mathbf{h}_j & \mathbf{1} + \mathbf{h}_i + \mathbf{h}_j & \mathbf{1} + \mathbf{h}_i + \mathbf{h}_j \\ \mathbf{1} + \mathbf{h}_i + \mathbf{h}_j & \mathbf{1} + \mathbf{h}_i + \mathbf{h}_j & \mathbf{h}_i & \mathbf{h}_j \end{bmatrix} \quad (16)$$

where $\mathbf{1}$ denotes the m -dimensional column vector whose components are all ones. Letting $n = 2p(p - 1)$, we define a $2m \times n$ matrix \mathbf{H} as

$$\mathbf{H} = [\mathbf{H}_{12}\mathbf{H}_{13} \cdots \mathbf{H}_{1p}\mathbf{H}_{23} \cdots \mathbf{H}_{2p}\mathbf{H}_{34} \cdots \mathbf{H}_{p-2,p}\mathbf{H}_{p-1,p}]. \quad (17)$$

Then it is easily seen that the $(n, n - 2m)$ code C having parity-check matrix \mathbf{H} is an SEC/DED/S4ED code [45], [46]. For example, in the case of $m = 4$, an SEC/DED/S4ED code of length 112 with eight check bits is constructed. Shortening this code by 40 bits, we have a $(72, 64)$ SEC/DED/S4ED code. In order to have 64 information bits, SEC/DED codes require at least eight check bits. The construction method described above gives a code having the same redundancy as ordinary SEC/DED codes and, in addition, the capability of detecting any errors in a single 4-bit byte.

As high-speed semiconductor memories become larger, more powerful error-correcting codes are needed. This has made SbEC/DbED (single b -bit byte error correcting/double b -bit byte error-detecting) codes practical. Such codes are constructed from RS codes or Bose-Chaudhuri-Hocquengham (BCH) codes over $\text{GF}(2^b)$. If each symbol of $\text{GF}(2^b)$ is expressed as a b -bit byte, then RS codes or 2^b -ary BCH codes having minimum distance $d_{\min} = 4$ are SbEC/DbED codes. In applying these codes to semiconductor memories, they are usually substantially shortened to fit the number of information bits to the respective application. In such cases, we can sometimes obtain more efficient codes by slightly modifying RS codes [47].

Table I shows the minimum number of parity-check bits for the best known SEC/DED codes, SEC/DED/SbED codes, and SbEC/DbED codes when the number of information bits is 32, 64, and 128 for $b = 4$ and 8 [48].

B. Error-Correcting Codes for Magnetic/Optical Discs

Error correction for magnetic-disc memories began with the application of Fire codes [49] in the 1960's. Later, RS codes

TABLE I
MINIMUM NUMBER OF PARITY-CHECK BITS FOR THE BEST KNOWN ERROR-CONTROL CODES FOR HIGH-SPEED MEMORIES

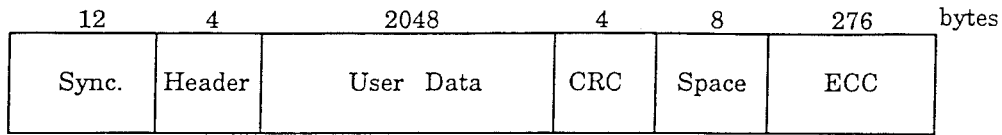
code	b	number of information bits		
		32	64	128
SEC/DED	-	7	8	9
SEC/DED/SbED	4	7	8	9
	8	10	10	11
SbEC/DbED	4	12	14	16
	8	24	24	24

assumed the major role in this area. On the other hand, RS codes have been the principal error-correcting codes used in optical-disc memories from the beginning. These applications were greatly expanded with the appearance of the compact disc (CD) and the CD-read-only memory (CD-ROM) at the beginning of the 1980's [46], [50].

The CD format uses a coding system called CIRC (cross-interleaved RS codes), which is a combination of two shortened RS codes of minimum distance $d_{\min} = 5$ over $\text{GF}(2^8)$. The information sequence is first encoded by a $(28, 24)$ code and then by a $(32, 28)$ code. For audio signals, an error interpolating technique can be used to estimate the original signal values by interpolating or extrapolating from reliable values for those errors that cannot be corrected.

However, such a technique is not applicable for data stored in computer systems, and thus greater error correction capability is required for the CD-ROM. Thus a CRC (cyclic redundancy check) code with 32 check bits and a product of two shortened RS codes with minimum distance $d_{\min} = 3$ over $\text{GF}(2^8)$ are used in addition to the CIRC. The encoder for a CD-ROM first appends four check bytes of a CRC to the user data of 2048 bytes with a 4-byte header and then adds eight bytes of all zeros for future extension. Alternate bytes are taken from the resulting sequence to produce two sequences of length 1032 bytes. Then, after rearranging the bytes, each sequence is encoded by a product of $(26, 24)$ and $(45, 43)$ RS codes to give a codeword of length 1170 bytes. Finally, the two codewords are combined and a synchronization pattern of 12 bytes is added to make one sector of a CD-ROM having 2352 bytes, as shown in Fig. 9, which is then applied to the CIRC encoder. A decoder for a CD-ROM is depicted in Fig. 10.

There are various ways of decoding the product code [51]. Reliability information for each symbol is obtained from the CIRC decoding process. Unreliable symbols are interpolated for audio CD's, but they are treated as erasures for CD-ROM's. If necessary, more detailed reliability information is available and soft-decision iterative decoding can be applied to make the decoding results as reliable as possible. However, the error rate of current CD's is not high, and the CIRC itself is an error-control coding scheme powerful enough for most CD's. Thus if the error probability of CIRC decoding is small enough, simple erasure decoding for the two RS codes can be adopted,



CRC: check bytes of CRC
 ECC: check bytes of two codewords of the product code

Fig. 9. The structure of one sector of a CD-ROM.

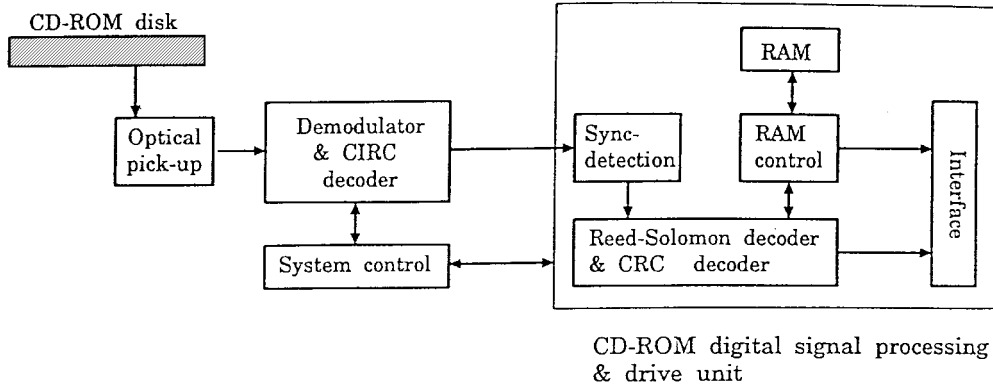


Fig. 10. The CD-ROM decoder.

as is the case for most CD-ROM drive units. With a single decoding of each of the two codes, all erasures in any eight or fewer positions can be corrected, since the minimum distance of the product code is $d_{\min} = 9$. But there exists at least one pattern of nine erasures that cannot be corrected by any decoder. If erasure decoding of the product code is iterated a few times, though, many patterns of nine or more erasures can be corrected.

Each of the RS codes in the CD and CD-ROM systems is decoded by directly calculating the coefficients of the error locator polynomial from the syndromes. This technique, which is simpler than other methods of decoding RS codes, can be applied whenever the minimum distance is no more than 7 or 8, as is the case for these RS codes.

Since the epoch-making success of the application of the CIRC system to CD's, the major error control roles for optical and magnetic discs have been monopolized by RS codes. This popularity is also supported by the progress of implementation techniques for codes with high error-correcting capability. Recently, it has become common to use RS codes over $GF(2^8)$ having very large minimum distances. For example, there exists a general purpose encoder/decoder chip for $(255, 255 - p)$ RS codes for which the number of check bytes p can be varied up to 36 [52]. This chip can correct up to s erasure bytes and t error bytes for any nonnegative integers s and t satisfying $2t + s \leq p$ at speeds of at least 33 Mbytes/s.

V. APPLICATIONS IN DIGITAL AUDIO/VIDEO TRANSMISSION

In this section, we present a specific example of an application of a cyclic code to digital audio broadcasting (DAB) and then discuss more generally the various coding systems used in DAB and digital video broadcasting (DVB) standards.

A. Difference Set Cyclic Codes for FM Multiplexed Digital Audio Broadcasting

There are many error-correcting codes that have been applied to DAB systems. Among them, the difference set cyclic code [53] adopted for an FM multiplexed DAB system is an example of an application involving a majority logic decodable code [54]. The initial use was for error correction in a TV teletext system [55]. For Western languages, the $(8, 4)$ extended Hamming code sufficed to keep the transmitted text readable. However, for the Japanese language this was not the case, because the language has ideograms called *kanji* (Chinese characters) and even one error may make the text unreadable or incorrect.

Hence more powerful codes were desired. However, the decoder had to be very simple, because it was to be installed in a TV set for home use. For these reasons, a $(273, 191)$ difference set cyclic code was adopted as a simply decodable code with high error-correction capability. This code has minimum distance $d_{\min} = 18$ and can correct eight or fewer errors. In practice, it was shortened by one bit to a $(272, 190)$ code to fit the data format of the teletext.

The $(273, 191)$ code is constructed on the basis of a perfect difference set

$$\begin{aligned}
 D &= \{d_0, d_1, \dots, d_{16}\} \\
 &= \{0, 18, 24, 46, 50, 67, 103, 112, 115, 126, \\
 &\quad 128, 159, 166, 167, 186, 196, 201\}.
 \end{aligned}$$

Any codeword $\mathbf{x} = (x_0, x_1, \dots, x_{272})$ in this code satisfies the following parity-check equations:

$$\begin{aligned}
 x_0 + x_{d_{i,0}} + \dots + x_{d_{i,i-1}} + x_{d_{i,i+1}} + \dots + x_{d_{i,16}} &= 0, \\
 i &= 0, 1, \dots, 16 \quad (18)
 \end{aligned}$$

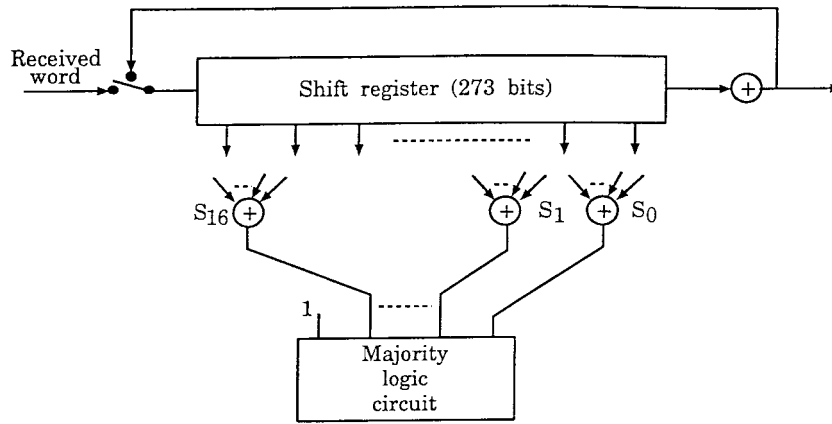


Fig. 11. Majority logic decoder for the (273, 191) code.

where $d_{ij} = d_i - d_j \bmod 273$, $0 \leq d_{ij} < 273$. Every symbol of a codeword except x_0 appears once and only once in these equations, and x_0 is contained in all the equations. Let r_j be the received symbol for the j th bit x_j of the transmitted codeword, $j = 0, 1, \dots, 272$. (r_j may be an analog symbol in the case of soft output demodulation.) We denote the result of a decision on r_j at a previous stage of decoding by $[r_j]$ and the error in $[r_j]$ by e_j , i.e., $e_j = x_j - [r_j] \bmod 2$, $e_j \in \{0, 1\}$. Initially, $[r_j]$ is taken to be the hard quantized version of r_j . Then, defining a syndrome bit s_i ($i = 0, 1, \dots, 16$) for the received word $\mathbf{r} = (r_0, r_1, \dots, r_{272})$ by

$$\begin{aligned} s_i &= [r_0] + [r_{d_{i,0}}] + \dots + [r_{d_{i,i-1}}] + [r_{d_{i,i+1}}] + \dots + [r_{d_{i,16}}] \\ &= e_0 + e_{d_{i,0}} + \dots + e_{d_{i,i-1}} + e_{d_{i,i+1}} + \dots + e_{d_{i,16}} \end{aligned} \quad (19)$$

we can correctly estimate the error e_0 in $[r_0]$ as the majority element of $S = (s_0, s_1, \dots, s_{16}, 1)$ when the number of errors is no more than eight, because at least ten components of S are 1 if $[r_0]$ is erroneous and at most eight components of S are 1 otherwise. The errors in the other bits are estimated in the same way after the received word is cyclically permuted. Fig. 11 shows a decoder based on this principle.

This majority logic decoder is much simpler than a BCH decoder of comparable length and error-correcting capability. In fact, a majority logic LSI decoder was fabricated with about one tenth the number of gates as a BCH decoder. For such a simple implementation, code efficiency must be compromised. However, considerable improvement in error-correcting capability is realized by introducing soft-decision decoding.

Optimal soft-decision decoding of the first bit x_0 in the transmitted codeword requires a MAP estimate of x_0 , or equivalently the error bit e_0 . In the remainder of this section, we consider MAP estimation of e_0 , i.e., estimating e_0 as the value that maximizes the *a posteriori* probability (APP) $P(e_0|\mathbf{r})$, given the received word \mathbf{r} . It is difficult to calculate $P(e_0|\mathbf{r})$ exactly. Therefore, some assumptions on the independence of the received symbols are needed. For an AWGN channel, the MAP estimate of e_0 is approximated as

$$\hat{e}_0 = \begin{cases} 1, & \sum_{i=0}^{16} s_i W_i > T \\ 0, & \sum_{i=0}^{16} s_i W_i < T \end{cases} \quad (20)$$

where \hat{e}_0 denotes the estimate of e_0 , T is a threshold given by

$$T = \frac{1}{2} \sum_{i=0}^{16} W_i, \quad (21)$$

and the weighting factor W_i is defined as

$$W_i = \frac{\log P_{ei}}{\log P_{oi}}. \quad (22)$$

P_{ei} and P_{oi} are the probabilities that the number of nonzero errors appearing in (19) is even and odd, respectively. This algorithm, which is called APP decoding [56], is near-optimal, but it is not practical because it demands too much complexity to compute W_i . Therefore, the following further approximation is commonly made [57].

Let \mathbf{e}_i be an error pattern corresponding to a previous decision concerning the received symbols appearing in (19), i.e.,

$$\mathbf{e}_i = (e_0, e_{j_1}, e_{j_2}, \dots, e_{j_{16}})$$

where

$$j_1 < j_2 < \dots < j_{16}$$

and

$$\{j_1, j_2, \dots, j_{16}\} = \{d_{i,0}, \dots, d_{i,i-1}, d_{i,i+1}, \dots, d_{i,16}\}.$$

We then approximate P_{ei} as the probability of the error pattern \mathbf{e}_{ei}^* having maximum probability among all \mathbf{e}_i of even weight. Similarly, we approximate P_{oi} by the probability of the error pattern \mathbf{e}_{oi}^* having maximum probability among all \mathbf{e}_i of odd weight. These \mathbf{e}_{ei}^* and \mathbf{e}_{oi}^* can be easily obtained by using the trellis shown in Fig. 12. Such an approximation of APP decoding for the difference set cyclic code is called APP trellis decoding.

The APP decoder is optimal if all 273-tuples $\mathbf{x} = (x_0, x_1, \dots, x_{272})$ satisfying (18) are codewords. Since this is not true in general, the decoding performance of APP decoding can sometimes be improved by variable threshold decoding [58], which involves iterative decoding with a variable threshold. At first, the threshold T is set high to avoid miscorrection. The decoding result for a received symbol then replaces the previous decision in succeeding stages of APP decoding. This operation gives so-called extrinsic information

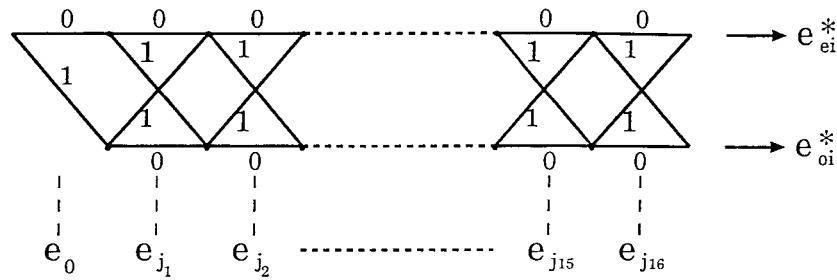


Fig. 12. The trellis for computing e_{ei}^* and e_{oi}^* .

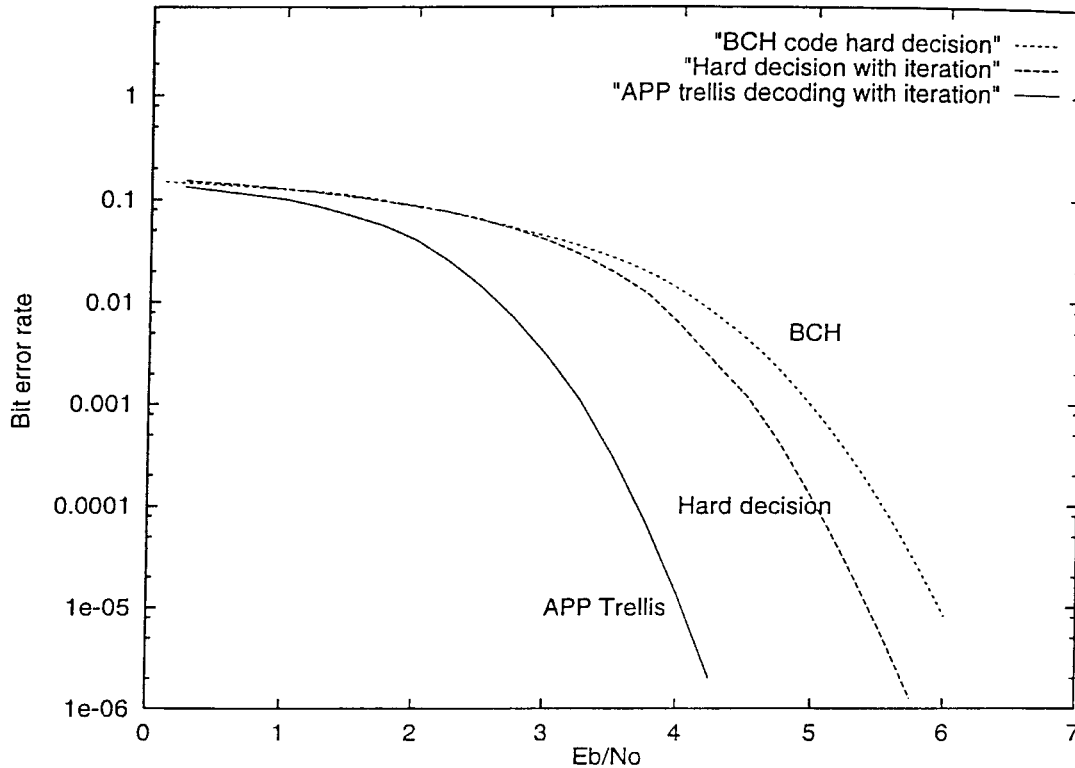


Fig. 13. Decoding performance of the (273, 191) code and a (273, 192) BCH code.

[32] to the decoding of the other symbols. After each iteration, T is decreased to correct additional bits. When T reaches the value given by (21), decoding is complete.

Fig. 13 shows the decoding performance of APP trellis decoding and hard decision decoding over an AWGN channel. In both cases, variable threshold decoding with three iterations was adopted. In the figure, the hard-decision decoding performance of a (273, 192) shortened BCH code correcting nine or fewer errors is also shown for comparison. Variable-threshold APP trellis decoding shows excellent decoding performance in spite of the simplicity of its implementation. A decoder chip for this algorithm has been implemented for application to FM multiplexed DAB systems.

B. Rate-Compatible Punctured Codes for Digital Audio Broadcasting

The European DAB System, which will be also used for DVB and the MPEG audio system [59], makes extensive use of rate-compatible punctured convolutional (RCPC) coding

schemes. The basic principles of these schemes are described below.

1) *Unequal and Adaptive Error Protection with RCPC Codes:* Rate $r = 1/n$ convolutional nonsystematic feedforward or systematic feedback encoders have the advantage of a binary trellis and therefore employ a simple Viterbi or MAP decoder [32]. From a (2, 1, 5) nonsystematic feedforward encoder with generator matrix

$$\mathbf{G}(D) = [1 + D + D^4, \quad 1 + D^3 + D^4] \quad (23)$$

we obtain a systematic feedback encoder with a rational parity generator

$$\mathbf{G}(D) = \left[1 \quad \frac{1 + D + D^4}{1 + D^3 + D^4} \right]. \quad (24)$$

Both encoders generate the same set of codewords, but they have slightly different BER performance. For very noisy channels, the BER curve of the feedback encoder is better than and approaches, but does not cross, the BER curve for uncoded performance.

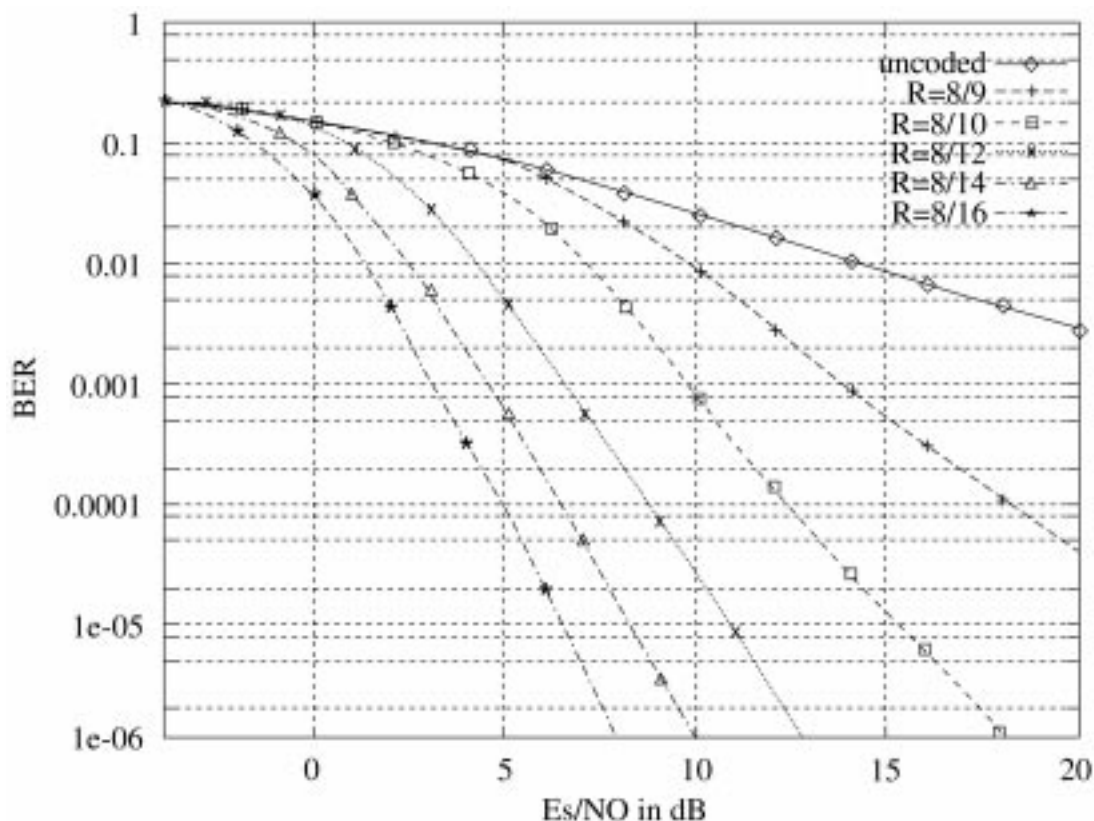


Fig. 14. BER performance of nonsystematic feedforward RCPC codes for $K = 5$ and $R = 8/9$ to $R = 1/2$ on the statistically independent Rayleigh fading channel.

Higher rate codes are easily obtained by puncturing. If this is done in a rate-compatible way, such that all the code bits in the higher rate code are also used for the lower rate codes, we obtain an RCPC code [60]. This code family, with a puncturing period of P , has rates equal to

$$r = \frac{P}{P+l}, \quad \text{for } l = 1 \text{ to } P \cdot (n-1). \quad (25)$$

The lowest rate is the rate of the original rate $r = 1/n$ code, called the mother code. Puncturing tables for different rates are given in [60]. Even lower rates can be achieved by duplicating bits, i.e., by replacing “1” by “2” or some higher integer in the puncturing tables. On a multiplicative, fully interleaved Rayleigh fading channel these systematic codes perform as shown in Fig. 14. By puncturing all of the nonsystematic bits, the rate 1 (uncoded) system is easily incorporated. Therefore, we have a great variety of coding options which can be easily adapted to source and channel requirements because the encoder and the decoder are the same for the whole code family and can be adapted using only the puncturing control. At the receiver, punctured bits marked with “0” in the puncturing tables are stuffed with zero values and repeated bits are combined by adding their soft received values. Therefore, the decoder always operates with the mother code and its trellis remains unchanged.

2) *The RCPC Coding Scheme for the European DAB System:* The European DAB system [61] uses the source-coding system of ISO/MPEG Audio [59]. In this case, PCM Audio at 2 times 768 kbps is compressed, typically down to 2 times 128 kbps

or less. Using a psychoacoustic model of the human ear, the audio band is divided into 32 subbands which employ adaptive quantization. The data stream sent to the multiplexer contains headers, a scalefactor and scalefactor-select information, and quantized samples with bits of different significance. This format and the dynamic data rate require unequal and adaptive error protection. Therefore, the RCPC codes described above have been designed for three different rates ($1/3$, $4/9$, and $8/15$) using a constraint length $K = 7$ mother code with 64 states. Later, in [62], an even more sophisticated system was proposed, where the rate pattern of the RCPC codes can be changed almost continuously within the data frame, depending on the varying rate and protection requirements of the source bits. As a result, the resource “protection bits” are used according to the needs of the source bits and the variation of the audio signal. The overall system exhibits a more graceful degradation at the edge of coverage of the radio broadcast system where the SNR deteriorates.

C. Error-Correcting Codes for Digital Video Systems

Decoding speed is at the top of the list of requirements for error control in digital video systems. For this reason, powerful error-correcting codes had not been considered for this application until the 1990’s. The first standardized error-correction method for digital video transmission, recommended as a part of the ITU-T H.261 standard in 1990, is very simple. It uses a (511, 493) BCH code, which has minimum distance $d_{\min} = 6$ and is capable of correcting two or fewer random errors and

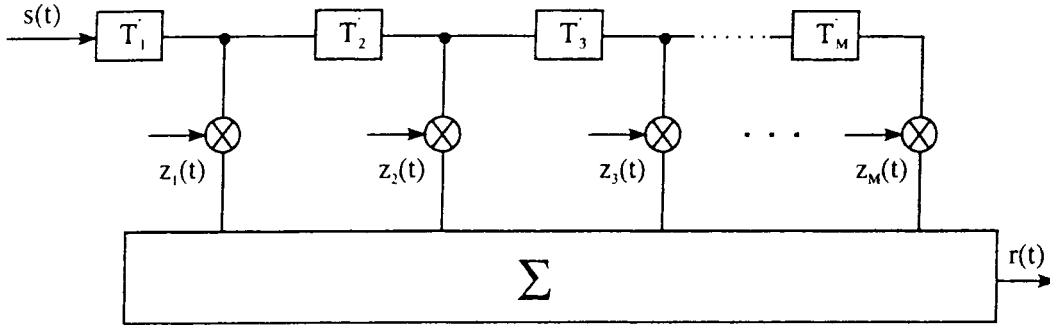


Fig. 15. Simplified model of the mobile communications channel.

detecting three random errors. In addition, it is designed to correct a burst error of length no longer than six.

Since then, however, more sophisticated error-correction schemes based on RS codes have been put into practical use with advanced video coding standards like MPEG2. For DVB, in particular, concatenated codes composed of an RS code as the outer code and a convolutional or trellis code as the inner code have been used. For example, a (204, 188) code obtained by shortening the (255, 239) RS code with minimum distance $d_{\min} = 17$ has been adopted as the outer code along with the (2, 1, 7) CCSDS convolutional code (sometimes punctured) as the inner code in the European digital terrestrial TV broadcasting system.

Multilevel coding with multistage decoding [63] is a strong candidate for future DVB systems. By employing unconventional signal labeling (mapping), the large number of nearest neighbors associated with multistage decoding can be reduced significantly. This results in highly efficient transmission for multivalued modulation formats such as M -PSK and M -QAM with moderate complexity [64]. Also, because of its excellent error performance, turbo coding [29] may be considered as the inner code in a concatenated system for future applications. Both turbo coding and multilevel coding suffer from relatively long decoding delays, but this might be allowed in some DVB applications.

VI. APPLICATIONS IN MOBILE COMMUNICATIONS

The advent of digital mobile communications during the last 15 years has resulted in the frequent use of error-control coding because the mobile channel exhibits raw error rates of 5 to 10%, mainly due to fades, bursts, and interference from other users and other cells. This also means that decoders must work at low values of E_b/N_0 . Consequently, all the features known to decoder designers must be used, such as interleaving, soft decisions, channel state information, and concatenation. On the other hand, bandwidth is scarce and expensive in a heavily booming cellular market. This means that the resource "redundancy" must be used with care, applying methods such as unequal and adaptive error protection. Of course, the dramatic increase in integrated circuit complexity has allowed the use of very sophisticated error-control methods. It is quite common to employ in a handheld mobile phone two Viterbi

algorithms in tandem, one for the equalizer and one for the FEC decoder, all on one chip, which also includes source compression for the speech signal, channel estimation, and other tasks.

In this section, we will give some Shannon theory background for the mobile channel, describe the most widely used FEC schemes in mobile communications, and conclude with some advanced FEC features currently being discussed for digital mobile systems.

A. The Mobile Radio Channel and the Shannon Limit

The mobile communications channel can be modeled as shown in Fig. 15, where the received complex equivalent baseband signal after multipath distortion is

$$r(t) = \sum_{m=1}^M s(t - T_m) \cdot z_m(t) \quad (26)$$

with complex time-varying factors $z_m(t)$. At the receiver we also have noise and interference, usually modeled as an additive AWGN component with one-sided power spectral density N_0 . If the channel is nonfrequency-selective, i.e., if the delay $T_M = \sum_{m=2}^M T_m$ is less than the symbol duration, then the channel induces only multiplicative distortion with $z = x + jy$. The amplitude

$$|z| = a = \sqrt{x^2 + y^2} \quad (27)$$

is then Rayleigh- or Rician-distributed. For the Rayleigh case we have

$$f_a(a) = \frac{a}{\sigma^2} \cdot e^{-a^2/2\sigma^2} \quad (28)$$

and a uniformly distributed phase. If sufficient interleaving is applied, these random variables become statistically independent. For coherently detected BPSK modulation, only the amplitude fluctuations are relevant. These assumptions allow us to calculate Shannon's channel capacity for this mobile channel. The capacity C is calculated as a function of the channel SNR

$$E_s/N_0 = rE_b/N_0 \quad (29)$$

where r represents the code rate. If we transmit at the capacity limit we obtain from the parameter equation

$$r = C(rE_b/N_0) \quad (30)$$

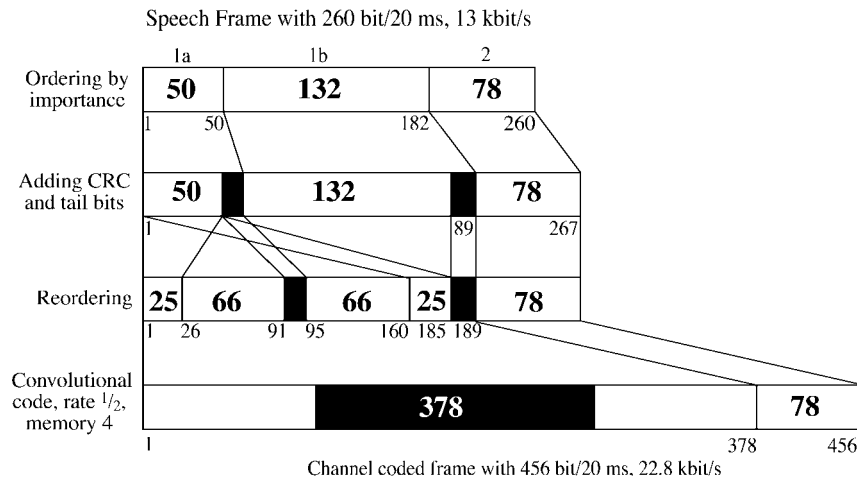


Fig. 16. FEC scheme for the full rate GSM speech coder.

TABLE II
SHANNON LIMIT FOR THE RAYLEIGH FADING CHANNEL

Input	Output	Rate	$E_b/N_0 _{\min}$ Rayleigh
binary	soft + CSI	$\rightarrow 0$	-1.6 dB
binary	soft	$\rightarrow 0$	-0.5 dB
binary	binary	$\rightarrow 0$	1.4 dB
binary	soft + CSI	0.5	1.8 dB
binary	soft	0.5	2.5 dB
binary	binary	0.5	4.9 dB

the Shannon limit

$$E_b/N_0|_{\min} = f(r). \tag{31}$$

For an interleaved Rayleigh fading channel, depending on the use of soft decisions and channel state information (CSI), the Shannon SNR limits of Table II can be calculated [65].

This Shannon theory result tells the designer of an FEC system for such a mobile channel that

- for low-rate codes, the mobile Rayleigh channel is not worse than the Gaussian channel, because low-rate interleaved coding acts like a multiple diversity system which transforms independent Rayleigh channels into a Gaussian channel;
- for rate 1/2 codes, the Rayleigh channel degrades the system by 2 to 3 dB, and
- soft decisions can gain up to 2.4 dB, and CSI, namely the fading amplitude, can gain roughly another decibel.

B. FEC Coding in the Global System for Mobile Communications (GSM)

With currently some 80 million mobile phones worldwide, the error-control system of GSM is probably the most widely used form of nontrivial FEC besides the compact disc and high-speed modems. The speech and data services of the GSM standard use a variety of FEC codes, including BCH codes and Fire codes, plus CRC codes for error detection and codes for synchronization, access, and data channels. These

codes are used as outer codes in a concatenated scheme. We will concentrate here on the inner coding scheme, which is primarily used for the full rate speech traffic channel. A 20-ms slot of the 13-kbit/s full rate speech coder produces a block of 260 bits, split into three sensitivity classes. 78 bits are rather insensitive to errors and are unprotected. The 50 most significant bits are initially protected by a 3-bit CRC used for error detection. Then these 53 bits, along with the remaining 132 bits, or a total of 185 bits, are encoded with a (2, 1, 5) nonsystematic convolutional code with generator matrix

$$G(D) = [1 + D^3 + D^4, 1 + D + D^3 + D^4]. \tag{32}$$

In order to terminate the trellis, four known tail bits are added, as shown in Fig. 16. This leads to a total of 378 coded bits and, including the 78 uncoded speech bits, results in a block of 456 bits every 20 ms, for a rate of 22.8 kbit/s. Together with seven other users, these bits are interleaved in groups of 57 bits and spread over eight TDMA bursts, which are transmitted over the mobile channel using GMSK modulation. The bursts include a midamble of 26 bits (+ two signaling bits) used by the receiver to estimate the channel tap values corresponding to the multipath model shown in Fig. 15. Using the channel tap estimates, an MLSE equalizer employing the Viterbi algorithm outputs hard or soft values for the 456 bits in a block. After deinterleaving, the Viterbi decoder accepts 378 values and outputs 185 bits from the terminated 16-state trellis. If the check with the CRC code is positive, the 50 plus 132 decoded bits, together with the 78 unprotected bits, are delivered to the speech decoder. A negative CRC usually triggers a concealment procedure in the speech decoding algorithm, rather than accepting any errors in the 50 most important bits.

When, as defined in the GSM standard, frequency hopping is used on the bursts in an urban environment, statistically independent Rayleigh fading is a reasonable assumption. In this case, we notice from the Shannon limits in Table II that at a rate of $r = 1/2$ we can gain up to 3.1 dB in required SNR if the channel decoder uses CSI and soft values. Since a normal Viterbi equalizer delivers only hard decisions, a modification is necessary leading to a SOVA or MAP type soft-in/soft-out

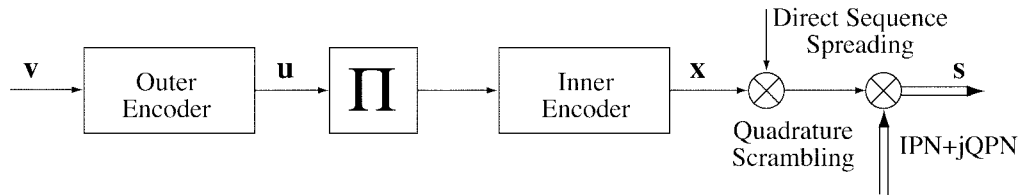


Fig. 17. A CDMA transmitter.

TABLE III
FEC CODES IN GSM SPEECH CODING

Service	Speech rate	constraint lengths	code rates
Full rate	13.0	5	1/2
Enhanced full rate	13.0	5	1/2
Half Rate	5.6	7	1/2, 1/3

equalizer. Most GSM terminals now use some kind of soft values from the equalizer as inputs to the decoder.

GSM is a good example how Shannon's theory can help a system designer. Although with such simple codes we are not operating at the absolute values of Shannon's limit given in Table II (GSM operates at approximately 7 dB), the relative gains between hard and soft decisions remain a good indicator.

Other FEC coding schemes used for the different speech options defined by GSM are summarized in Table III. For data services, the currently defined user data rates in GSM are 2.4, 4.8, and 9.6 kbit/s. When a full rate channel is used the respective code rates are 1/6, 1/3, and 1/2. It will be a difficult task to accommodate higher user data rates in one full rate slot.

C. FEC Coding in the CDMA Mobile System (IS-95)

In CDMA systems for mobile communication, FEC is very important in combating multiuser interference (MUI) and the influence of multipath fading. Therefore, most systems use a low-rate channel code followed by direct sequence (DS) spreading with a high processing gain. Others use a concatenation of two or more low-rate codes followed by DS spreading, which provides privacy (a signature sequence), but has a low processing gain. A well-known example of the latter type of system is the uplink of the Qualcomm system based on the IS-95(A) standard [66]. As shown in Fig. 17, this system consists of an outer convolutional code C^O with rate $r^O = 1/3$ and constraint length $K = 9$, a block interleaver (Π) with 32 rows and 18 columns, and a 64-ary orthogonal modulation scheme. The orthogonal modulation (the inner code C^I) is a so-called Hadamard code, or Walsh function, and can be viewed as a systematic block code of rate $r^I = 6/64$ and minimum distance $d_{\min} = 32$. The subsequent processing steps are: 1) DS spreading by a factor of 4, 2) quadrature scrambling of the in- and quadrature-phase components with different PN sequences (IPN and QPN), and 3) offset QPSK modulation (not shown in Fig. 17).

In the following, we describe two types of receiver structures. The first is a coherent receiver, where we must know

the carrier phase, the delays of the multipath channel, and the complex factors of the different propagation paths (i.e., perfect CSI estimation). This leads to an M -finger RAKE receiver with *maximum ratio combining* (MRC) [67]. In each RAKE finger, the respective path delay is compensated, followed by a multiplication with the complex conjugate propagation coefficient z_m .

Classical demodulation after the coherent RAKE receiver involves maximum-likelihood (ML) detection of the transmitted Hadamard codeword by searching for the maximum value of the correlation vector, which is easily implemented with the fast Hadamard transform (FHT). The systematic bits (information bits) of the demodulated codeword are then weighted with the maximum value of the correlation vector. (The Hadamard code can also be decoded with a "soft-in/soft-out" decoder [32].) Afterwards, they are passed on to the deinterleaver and the outer soft-in/hard-out Viterbi decoder.

The other case is a noncoherent receiver design without any knowledge of the phase of the signal. Here, in each RAKE finger, the FHT must be performed for both the in-phase and quadrature-phase components after the delay compensation. After *square-law combining* (SLC) [67] of the $2M$ correlation vectors, classical ML detection is also performed to find the maximum value of the correlation vector. Further decoding of the outer code is straightforward, as in the coherent case.

D. Relations Between Source and Channel Coding for Mobile Channels

A basic result of Shannon's information theory is that source and channel coding can be treated separately. Of course, this is true in the information-theoretic sense: as long as the entropy of the source is less than the channel capacity, there exists a separable source and channel coding scheme that allows transmission over the channel with arbitrarily low error probability. Arising from this basic result, there has been a clear-cut interface between source and channel coding and decoding. However, on a transmission channel with extensive noise, interference, multipath, fading, and shadowing, it is too expensive in terms of bandwidth, delay, and complexity to implement perfect channel coding.

In a practical system, the different blocks of source coder, channel coder, transmission channel, channel decoder, and source decoder are linked in a variety of ways, as indicated in Fig. 18.

- Source significance information (SSI) is passed to the channel coder for static or dynamic unequal error protection. Unequal error protection can be easily performed

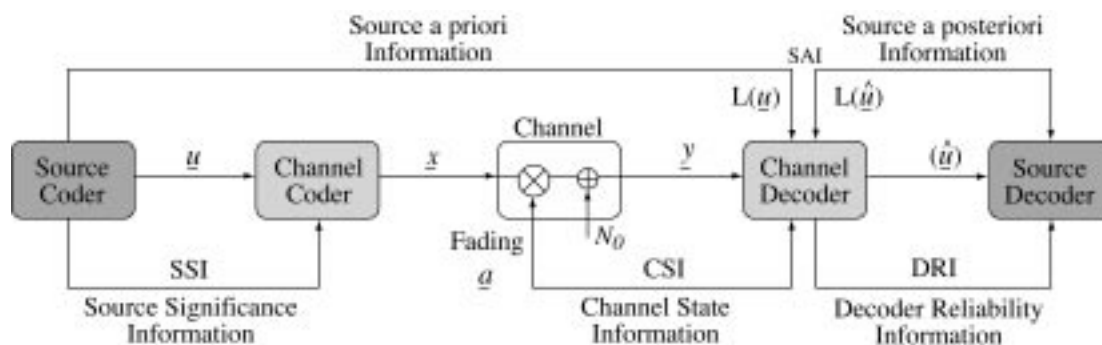


Fig. 18. Links between source and channel coding in a transmission over a mobile channel.

with block and convolutional codes, and in a very simple way by RCPC codes, as described in Section VII-B1).

- The channel decoder and the demodulator are not only connected by the hard decisions of the demodulator/detector. Soft decisions and channel state information (CSI) are also passed on.
- At the receiver, when the channel decoder makes errors and these errors can be detected by a CRC or by other means, it is common engineering practice to interpolate the source sample values. One can also use more sophisticated methods to conceal errors by using the decision on the current bit and its reliability. This sophisticated concealment, however, requires channel decoders that deliver “soft outputs,” i.e., decisions and “decision reliability information” (DRI) as shown in Fig. 18.
- Another approach links source and channel decoding to an even greater extent. Source-controlled channel decoding [68] uses *a priori* and *a posteriori* information about the source bits. For reasons of delay and complexity and because of highly nonstationary sources, many source coding schemes still contain redundancy and some bits are highly correlated, at least at times. Under these circumstances source and channel decoding should be linked more closely. In fact, Shannon mentioned this possibility in his 1948 paper [1]: “However, any redundancy in the source will usually help if it is utilized at the receiving point. In particular, if the source already has redundancy and no attempt is made to eliminate it in matching to the channel, this redundancy will help combat noise.” Consequently, if some redundancy or bit correlation is left by the source coder, this should be utilized jointly by the channel and source decoder. The source *a priori/a posteriori* information (SAI), as indicated in Fig. 18, tells the decoder with a certain probability the value of the next bit to be decoded. It is much better to give the channel decoder all the information known about the bits which are to be decoded, i.e., about correlation or bias values, than to allow it to make errors by blind decoding, thus causing it to conceal these errors later. The goal of source-controlled channel decoding is to reduce the channel decoder error rate by supplying the channel decoder with source information. Especially on mobile channels, gains of several decibels in channel SNR have

been achieved for speech, audio, still image, and video sources [68], [69], [70].

The last three items can be implemented in existing systems because they do not require changes at the transmitter. They are “value-added” devices to be used only when appropriate, for instance, in a bad channel environment. An example for GSM can be found in [69].

E. Turbo Decoding for Mobile Communications

In 1993, decoding of two or more product-like codes was proposed using iterative (“turbo”) decoding [29]. (See also the end of Section II.) The basic concept of this new coding scheme is to use a parallel concatenation of at least two codes with an interleaver between the encoders. Decoding is based on alternately decoding the component codes and passing the so-called *extrinsic information* to the next decoding stage. For good soft-in/soft-out decoders this *extrinsic information* is an additional part of the soft output. Even though very simple component codes are used, the “turbo” coding scheme is able to achieve performance close to Shannon’s bound, at least for large interleavers and at BER’s of approximately 10^{-5} .

Moreover, it turned out that the turbo method originally applied to parallel concatenated codes is much more general [71] and can be successfully applied to many detection/decoding problems such as serial concatenation, equalization, coded modulation, multiuser detection, joint source and channel decoding, and others. We will mention here two applications for mobile communications to show the potential of turbo decoding.

1) *Parallel Concatenated (Turbo) Codes for GSM Applications:* The turbo decoding method was not available at the time of GSM standardization. But it is interesting to consider what can be gained in the GSM System with turbo decoding. In [72], the FEC system for GSM was redesigned with parallel concatenated codes, while keeping all the framing and the interleaver as in the full rate GSM System. This means that a frame length of only 185 information bits could be used. This is different from the common usage of turbo codes, where several thousand bits are usually decoded in one frame. Nevertheless, it was shown that on GSM urban channels with frequency hopping, an improvement in channel SNR (i.e., an additional coding gain) of 0.8 to 1.0 dB is possible. Although several iterations of the turbo decoder are necessary to achieve

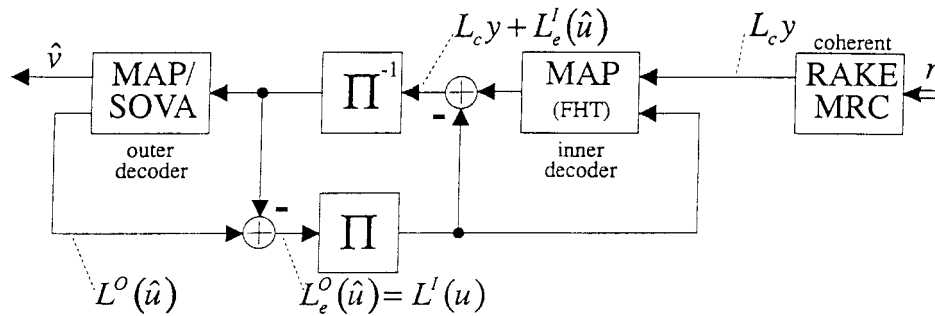


Fig. 19. Iterative decoding in a coherent receiver.

this gain, the total decoding complexity is comparable to the GSM system because the turbo component codes had only four states, compared to the 16-state code used in standard GSM.

2) *Turbo Detection in the IS-95 Mobile System:* In this section, we explain the principle of turbo detection for the IS-95 uplink when coherent demodulation is applied. An application to the noncoherent uplink can be found in [73]. For iterative decoding, it is necessary to replace the outer Viterbi decoder with a modified soft-output Viterbi algorithm (SOVA) or a MAP decoder which delivers soft-output information with each decoded bit. The soft information from the outer decoding stage is interleaved and then fed back as *a priori* information in a redecoding of the inner code. This leads to a serial turbo decoding scheme. A schematic overview of the receiver is given in Fig. 19.

From the L -finger RAKE receiver we obtain a vector \mathbf{y} . The received log-likelihood values $L(x|y) = L_c y + L_e^I(\hat{u})$, including the fed back *a priori* values $L^I(u)$ of the systematic bits, are correlated with the Hadamard codewords using the FHT. This results in the correlation vector \mathbf{w}' . The approximate MAP decoding rule then reduces to a simple expression

$$L^I(0) \simeq \frac{1}{2} \cdot \max_{j, u=+1} (w'_j) - \frac{1}{2} \cdot \max_{j, u=-1} (w'_j). \quad (33)$$

To obtain the *a priori* information, we must decode the outer code with a SOVA or MAP decoder. The extrinsic part of the soft output for the outer decoded bits is then used as the *a priori* information $L^I(u)$ for the systematic bits of the inner code in the feedback loop shown in Fig. 19.

Simulation results for the AWGN channel show that an additional coding gain of about 1.0–1.5 dB can be achieved with iterative decoding after five iterations. If the simple approximation in (33) is applied, the resulting degradation is less than 0.1 dB.

VII. FILE TRANSFER APPLICATIONS

File transfer is the movement of digital information from one data terminal to another. This digital information is organized into characters, frames, and files. File transfer applications typically demand a very high level of reliability; unlike voice applications, a single erroneous bit can render a megabyte file useless to the end user. Communication links in wireless and wired file transfer systems are designed to be highly reliable. In many systems, the majority of data errors

result from single-event upsets caused by equipment transients. The error-control systems for these applications use error detection coupled with retransmission requests to maximize reliability at some cost to throughput. At their simplest, such error-control systems use parity-check bits or CRC codes to trigger retransmission requests. Since the retransmission requests occur well below the application layer, and are in a sense “automatic,” such protocols are usually called *automatic repeat request* (ARQ) protocols. (The “RQ” in “ARQ” is taken from the Morse code designation for a retransmission request.) More complicated protocols include elements of FEC and packet combining to reduce the frequency of retransmission requests.

This section begins with a quick look at parity, CRC, and “pure” ARQ protocols. Several popular file-transfer protocols are provided as examples. Consideration is then given to hybrid protocols and packet combining.

A. Parity Bits and CRC's

In asynchronous file-transfer systems, each character is treated as a separate entity [74]. Seven-bit ASCII is the most ubiquitous example (where ASCII stands for “American Standards Committee for Information Interchange”). The 128 possible characters cover upper and lower case Roman letters, Arabic numerals, various punctuation marks, and several control characters. The standard error-control technique for asynchronous ASCII is the simplest possible: a single parity bit is appended to each ASCII character. The value of the appended bit is selected so that the sum of all eight transmitted bits is 0 modulo 2 (even parity) or 1 modulo 2 (odd parity). In either case, the resulting code can detect all single errors, as well as a large number of error patterns of higher weight. The encoding and decoding circuit is extremely simple, and can be constructed using a small number of exclusive-OR gates or a few lines of code. The *Kermit* data modem protocol is a typical example of an asynchronous file-transfer system that uses simple parity checks.

In synchronous transmission systems, error control can be expanded to cover entire frames of data. Since the frame is a larger transmission element than the character, the number of encoding and decoding operations is reduced, along with the number of potential requests for retransmission.

The most popular synchronous error-control techniques are based on shortened linear cyclic codes. Let C be an (n, k)

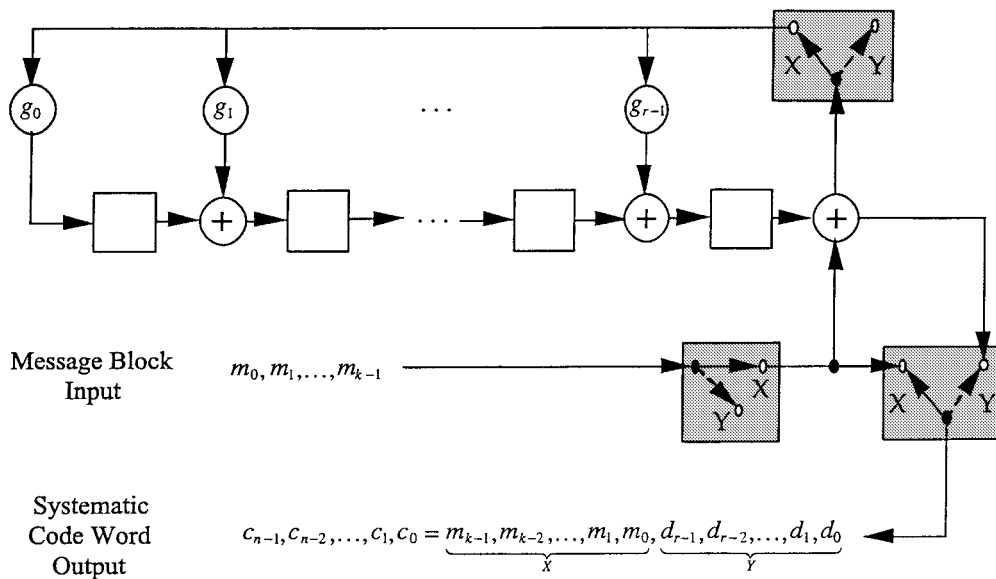


Fig. 20. A CRC encoder circuit.

cyclic code with a systematic encoder. Let S be the subset of codewords in C whose j high-order information coordinates (i.e., the j rightmost coordinates in the codeword) have the value zero. Let C' be the set of words obtained by deleting the j rightmost coordinates from all of the words in S . C' is an $(n - j, k - j)$ shortened systematic code that is usually referred to as a CRC, or *cyclic redundancy check*, code [75]. CRC codes are also called *polynomial codes*.

Note that CRC codes obtained from cyclic codes through shortening are almost always noncyclic. However, it is still possible to use the same shift-register encoders and decoders that are used with the original cyclic code. Consider the systematic cyclic encoder in Fig. 20. Before encoding begins, the shift-register memory elements are set to zero. If the first j high-order message symbols are equal to zero, their insertion into the encoding circuit does not change the state of the shift register, and results only in the output of j zeroes. The deletion of j high-order symbols from the message block thus has no impact on the encoding process except for the deletion of j zeroes in the corresponding message positions of the codeword.

The shift-register taps $\{g_0, g_1, \dots, g_{r-1}\}$ in Fig. 20 are the coefficients of a degree- r CRC generator polynomial

$$g(x) = x^r + g_{r-1}x^{r-1} + \dots + g_1x + g_0.$$

Given a message block

$$\mathbf{m} = (m_0, m_1, \dots, m_{k-1})$$

and the associated message polynomial

$$m(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \dots + m_1x + m_0$$

the encoder computes the remainder $d(x)$ obtained from the division of $x^r m(x)$ by $g(x)$. The remainder is appended to

the message bits to complete the systematic encoding. The resulting code polynomial has the form $c(x) = x^r m(x) + d(x)$.

The encoder can also be used as a decoder. Note that, by construction, $c(x) = x^r m(x) + d(x)$ must be divisible by $g(x)$ without remainder. If the encoded data is not corrupted during transmission, the “encoding” of the received version of $c(x)$ will result in a zero remainder. The detection of a nonzero remainder after the processing of the received data is the trigger for a retransmission request.

There are a variety of “rules of thumb” for selecting CRC generator polynomials. Bertsekas and Gallager [76] note that it is common practice to select $g(x) = (x + 1)b(x)$, where $b(x)$ is primitive. The $(x + 1)$ factor ensures that all odd-weight error patterns are detectable. Table IV contains several CRC codes of varying length that have been used in practice. Detailed examinations of various polynomials with different degrees are provided in [77]–[79].

The error-detecting capability of a CRC code is best measured in terms of the percentage of possible error patterns that are detectable. Channel errors tend to occur in bursts in the wired file-transfer applications that make the most use of CRC codes, so the memoryless channel assumption that underlies a Hamming distance analysis is not valid. An r -bit CRC can detect $100 \cdot (1 - 2^{-r})\%$ of all error patterns.

CRC codes are often altered in practice to avoid the possibility of confusing the check bits with protocol flag fields (frame delimiters, for example) or to maintain a minimum transition density (an important consideration in magnetic recording applications). The modification to the code typically involves complementing some or all of the check bits. Though the code is no longer linear, error-detection performance is unaffected.

In some applications it is easier to generate an r -bit checksum by simply segmenting the information to be protected into r -bit words and computing the sum (without carry). As with the unmodified CRC, the resulting code is linear and can detect $100 \cdot (1 - 2^{-r})\%$ of all error patterns.

TABLE IV
SEVERAL CRC GENERATOR POLYNOMIALS THAT ARE COMMONLY USED IN PRACTICE

CRC Code	Generator Polynomial
CRC-4	$g_4(x) = x^4 + x^3 + x^2 + x + 1$
CRC-7	$g_7(x) = x^7 + x^6 + x^4 + 1 = (x^4 + x^3 + 1)(x^2 + x + 1)(x + 1)$
CRC-8	$g_8(x) = (x^5 + x^4 + x^3 + x^2 + 1)(x^2 + x + 1)(x + 1)$
CRC-12	$g_{12}(x) = x^{12} + x^{11} + x^3 + x^2 + x + 1 = (x^{11} + x^2 + 1)(x + 1)$
CRC-ANSI	$g_{ANSI}(x) = x^{16} + x^{15} + x^2 + 1 = (x^{15} + x + 1)(x + 1)$
CRC-CCITT	$g_{CCITT}(x) = x^{16} + x^{12} + x^5 + 1$ $= (x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1)(x + 1)$
CRC-SDLC	$g_{SDLC}(x) = x^{16} + x^{15} + x^{13} + x^7 + x^4 + x^2 + x + 1$ $= (x^{14} + x^{13} + x^{12} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x + 1)(x + 1)^2$
CRC-24	$g_{24}(x) = x^{24} + x^{23} + x^{14} + x^{12} + x^8 + 1$ $= (x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x + 1)(x^{10} + x^9 + x^6 + x^4 + 1)$ $\cdot (x^3 + x^2 + 1)(x + 1)$
CRC-32 _A [80]	$g_{32_A}(x) = x^{32} + x^{30} + x^{22} + x^{15} + x^{12} + x^{11} + x^7 + x^6 + x^5 + x$ $= (x^{10} + x^9 + x^8 + x^6 + x^2 + x + 1)(x^{10} + x^7 + x^6 + x^3 + 1)$ $\cdot (x^{10} + x^8 + x^5 + x^4 + 1)(x + 1)(x)$
CRC-32 _B [78]	$g_{32_B}(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
CRC-32 _C [76]	$g_{32_C}(x) = x^{32} + x^{26} + x^{23} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

In the next section we consider the various means by which retransmission requests can be incorporated into a file-transfer protocol.

B. Retransmission Request Modes

In this section we derive the basic relations and design choices that determine the throughput performance of an ARQ protocol. We begin by determining the number of times that a given packet can expect to be transmitted before it is accepted by the receiver. Given the probability P_r that an error is detected and a retransmission request generated, the expected number of transmissions T_r is computed as follows:

$$\begin{aligned} T_r &= (1 - P_r) + 2P_r(1 - P_r) + 3P_r^2(1 - P_r) + \dots \\ &\quad + kP_r^{k-1}(1 - P_r) + \dots \\ &= (1 - P_r) \sum_{k=1}^{\infty} kP_r^{k-1} = \frac{1}{1 - P_r}. \end{aligned} \quad (34)$$

There are three basic modes that are used in the implementation of retransmission-request protocols: stop-and-wait (SW-ARQ), go-back-N (GBN-ARQ), and selective-repeat (SR-ARQ) [75], [80], [81]. These three modes provide varying levels of throughput by changing the amount of buffering required in the transmitter and/or receiver. Throughput (η) is defined here to be the average number of encoded data blocks (packets) accepted by the receiver in the time it takes the transmitter to send a single k -bit data packet.

SW-ARQ is by far the simplest of the three modes. In this mode, the transmitter sends a single packet and waits for an acknowledgment (ACK). If an ACK is received, the transmitter sends the next packet. Otherwise, the first packet is retransmitted. Since the transmitter is idle while waiting for the response, this mode of ARQ is often called "Idle RQ" [74]. Let Γ be the number of bits that the transmitter could have transmitted during this idle time. An SW-ARQ protocol based on a rate $R = k/n$ error-detecting code will provide the

following throughput:

$$\eta_{SW} = \frac{k}{T_r(n + \Gamma)} = R \left(\frac{1 - P_r}{1 + \Gamma/n} \right). \quad (35)$$

SW-ARQ protocols require very little buffering in the transmitter, and virtually none in the receiver. For this reason such protocols are often used in simple data modems (the "Kermit" protocol is a typical example). The disadvantage, of course, lies in the inefficient use of the channel, an inefficiency that increases with the distance between transmitter and receiver.

GBN-ARQ protocols assume that the transmitter is capable of buffering N packets. When the transmitter is informed by the receiver of an error in some packet, say packet j , the transmitter "goes back" to that packet and resumes transmission from there. The parameter N is the number of packets that have been subsequently sent by the transmitter since the j th packet, and that must now be retransmitted. N is a function of the roundtrip propagation and processing delays for the transmitter and receiver and can be approximated by $\lceil \Gamma/n \rceil$. The throughput for a GBN-ARQ protocol based on a rate $R = k/n$ error-detecting code is as follows:

$$\eta_{GBN} = \left(\frac{k}{n} \right) \left(\frac{1}{1 + (T_r - 1)N} \right) = R \left(\frac{1 - P_r}{1 + P_r(N - 1)} \right). \quad (36)$$

Note that the transmitter in a GBN-ARQ protocol is continuously transmitting. GBN-ARQ protocols are thus often called "Continuous RQ" protocols [74].

SR-ARQ protocols are also Continuous RQ protocols. In this case the transmitter only retransmits the specific packets for which an ACK is not received. This requires significant buffering in both the transmitter and the receiver, but maximizes channel efficiency. The throughput in this case is no longer a function of channel delay. The throughput for a rate $R = k/n$ code is quickly computed as

$$\eta_{SR} = \left(\frac{k}{n} \right) \left(\frac{1}{T_r} \right) = R(1 - P_r). \quad (37)$$

In the next section we consider the means by which error detection retransmission requests are incorporated into two popular file-transfer applications.

C. High-Level Data-Link Control Protocol

The High-Level Data-Link Control (HDLC) protocol is a general-purpose protocol that operates at the data link layer of the OSI reference model [82]. It was created by the International Standards Organization (ISO) and is designated ISO 4335. HDLC uses the services of a synchronous physical layer and can provide a best effort or a reliable communications path between the transmitter and receiver (i.e., with acknowledged data transfer). The selected HDLC mode determines the type of service. HDLC is generally implemented using a Continuous RQ mode.

Link Access Procedure Version B (LAPB) is a subset of HDLC that is commonly used to control the transfer of information frames across point-to-point data links in public and private packet-switching networks. Such networks are generally referred to as X.25 networks [74].

The HDLC frame format has six basic elements [82].

- An 8-bit opening flag.
- An 8-bit address field (expandable in 8-bit increments).
- An 8-bit control field (expandable in 8-bit increments).
- An information field of arbitrary size.
- A 16-bit frame check field (32 bits optional).
- An 8-bit closing flag.

The bits in the 16-bit frame check field are computed using CRC-CCITT, as defined in Table IV. CRC-32_C is the optional 32-bit code that is used in applications requiring an extremely high level of reliability (e.g., high-speed local-area networks).

In full-duplex mode, HDLC allows for the use of “piggy-back acknowledgments”—a situation in which acknowledgments or requests for retransmission of packets on the forward channel are combined with other data transmitted on the return channel.

D. The Internet and TCP/IP

The key technology underlying the Internet (and many internets, intranets, and variations thereof) is the suite of protocols that go by the name *TCP/IP*. TCP/IP is named after two constituent protocols, the Transmission Control Protocol and the Internet Protocol, but the suite actually contains several dozen protocols [83]. TCP/IP allows for the internetworking of computer networks that are based on different networking technologies. For example, TCP/IP allows for the seamless interconnection of token rings (IEEE 802.5) with Ethernet-based systems (IEEE 802.3).

A TCP/IP internetworking architecture can be divided into four basic layers.

- An *Application Layer* consisting of user processes (TELNET, FTP, SMTP, etc.).
- A *Transport Layer* that provides end-to-end data transfer.
- An *Internetwork Layer* (also called the Internet layer) that provides the image of a single virtual network to upper layers by routing datagrams among the interconnected networks.

- A *Network Interface Layer* (also called the link or data-link layer) that provides the actual interface to the hardware of the individual networks (e.g. X.25, ATM, FDDI, and packet radio networks).

The Internet Protocol (IP) is the heart of the Internetwork layer. It is a connectionless protocol that does not assume or provide any measure of link reliability. The only error control incorporated into the IP datagram is a 16-bit checksum that only covers the header. The checksum is obtained by computing the 16-bit one’s complement of the one’s complement sum of all 16-bit words in the header (the checksum itself is assumed to be zero during this computation) [84]. If a received IP datagram has an incorrect checksum, the entire datagram is discarded. There is no provision for a retransmission request, in part because it would not be known from whom to request the retransmission. It should also be noted that IP does not provide a check on the data field within the datagram. This must be provided at the Transport layer.

There are two basic transport-layer protocols: the Transmission Control Protocol (TCP—connection oriented) and the User Datagram Protocol (UDP—connectionless). Both protocols “encapsulate” IP datagrams by appending transport-layer headers. UDP has an optional 16-bit checksum that, when used, is calculated in a manner similar to the IP datagram checksum. The UDP checksum is a 16-bit one’s complement of the one’s complement sum of the encapsulated IP header, the UDP header, and the UDP data. As with IP, UDP does not allow for acknowledgments. UDP simply arranges received packets in the proper order and passes them on to the application. Internet telephony is typical of the applications that use UDP in that it emphasizes latency over reliability.

TCP, on the other hand, makes full use of its 16-bit checksum. TCP assigns sequence numbers to each transmitted data block, and expects a positive acknowledgment from the transport layer of the receiving entity. If an acknowledgment is not received in a timely manner, the block is retransmitted. The full retransmission protocol is a variation of SR-ARQ that uses a window to control the number of pending acknowledgments. The transmitter is allowed to transmit all packets within the window, starting a retransmission clock for each. The window is moved to account for each received acknowledgment.

E. Hybrid-ARQ Protocols and Packet Combining

In 1960, Wozencraft and Horstein [85], [86] described and analyzed a system that allowed for both error correction and error detection with retransmission requests. Their system, now known as a type-I hybrid-ARQ protocol, provides significantly improved performance over “pure” ARQ protocols. The goal in designing such systems is to use FEC to handle the most frequently occurring error patterns. The less frequently occurring (those most likely to cause FEC decoder errors) are handled through error detection and the request of a retransmission.

Error-control protocols based on algebraic block codes (e.g., RS and BCH codes) and hard-decision decoding provide a natural form of type-I hybrid-ARQ protocol. The most

commonly used hard-decision decoders for BCH and RS codes use bounded distance decoding algorithms that correct a number of errors up to some design distance t . If there is no codeword within distance t of the received word, the bounded distance decoding algorithm fails, providing an indication of uncorrectable errors. When decoder failures are used as a trigger for a retransmission request, the BCH- and RS-based systems become examples of type-I hybrid-ARQ protocols [87]. The ReFLEX and cellular digital packet data (CDPD) wireless data protocols are typical of the applications that use these strategies.

Type-I hybrid-ARQ protocols can also be based on complete decoders. Drukarev and Costello developed the "Time-Out" and "Slope Control" algorithms for type-I protocols based on convolutional codes with sequential decoding [88]. Yamamoto and Itoh [89] developed a similar protocol based on Viterbi decoding.

Pure ARQ and Type-I hybrid-ARQ protocols generally discard received packets that trigger retransmission requests. In 1977, Sindhu [90] discussed a scheme that made use of these packets. Sindhu's idea was that such packets can be stored and later combined with additional copies of the packet, creating a single packet that is more reliable than any of its constituent packets. Since 1977 there have been innumerable systems proposed that involve some form of packet combining. These packet-combining systems can be loosely arranged into two categories: code-combining systems and diversity-combining systems.

In code-combining systems, the packets are concatenated to form noise-corrupted codewords from increasingly longer and lower rate codes. Code combining was first discussed in a 1985 paper by Chase [91], who coined the term. Subsequent code-combining systems are exemplified by the work of Krishna, Morgera, and Odoul [92], [93]. An early version of a code-combining system was the type-II hybrid-ARQ protocol invented by Lin and Yu [94]. The type-II system allows for the combination of two packets, and is thus a truncated code-combining system. The system devised by Lin and Yu was developed for satellite channels. Type-II systems based on RS and RM codes were later developed by Pursley and Sandberg [95] and by Wicker and Bartz [96], [97], and similar systems which utilize RCPC codes were introduced by Hagenauer [60].

In diversity-combining systems, the individual symbols from multiple, identical copies of a packet are combined to create a single packet with more reliable constituent symbols. Diversity-combining systems are generally suboptimal with respect to code-combining systems, but are simpler to implement. These schemes are represented by the work of Sindhu [90], Benelli [98], [99], Metzner [100], [101], and Harvey and Wicker [102].

VIII. CONCLUSION

In this paper, we have presented a number of examples illustrating the way in which error-control coding, an outgrowth of Shannon's Information Theory, has contributed to the design of reliable digital transmission and storage systems over the past 50 years. Indeed, coding has become an integral part

of almost any system involving the transmission or storage of digital information. And as we look to the future and the "digital revolution," with such innovations as digital cellular telephony, digital television, and high-density digital storage, it seems assured that the use of coding will become even more pervasive.

In trying to assess the impact that various advances in coding theory have had in the practical arena, some general trends become apparent. Although most of the early work in coding was based on algebraic constructions and decoding methods, recent trends definitely favor probabilistic, soft-decision decoding algorithms. With a few notable exceptions, such as the ubiquitous Reed-Solomon codes, the 2-3-dB loss associated with hard-decision decoding can no longer be tolerated in most applications. In fact, one of the most exciting current areas of coding research is the development of soft decoding methods for algebraically constructed block codes with good distance properties. Because of the relative ease of soft-decision decoding of convolutional codes (due to the development of sequential decoding, and later the Viterbi algorithm), convolutional codes became the preferred choice in most practical applications. The relative scarcity of nearest neighbor codewords in convolutional codes compared to block codes also tipped the scale in their favor, particularly in applications requiring only moderate BER's, where the number of nearest neighbors plays an important role in performance. On the other hand, in applications where data integrity is paramount, or when there is a strong tendency towards burst errors, powerful nonbinary error-correcting (e.g., Reed-Solomon) or binary error-detecting (e.g., CRC) block codes are still preferred.

Another noticeable trend is the relatively recent emergence of many commercial applications of coding. It is fair to say that the driving force behind almost all coding applications in the first 25 years after the publication of Shannon's paper was government scientific (e.g., NASA) programs or military applications. However, in the most recent 25 years, particularly with the emergence of the compact disc, high-speed data modems, and digital cellular telephony, commercial applications have proliferated. From our perspective, it seems fairly certain that this trend will not only continue but accelerate. Indeed, in this paper we have been able to touch on only a few of the many current activities involving the development of coding standards for various commercial applications. Thus it is clear that error-control coding has moved from being a mathematical curiosity to being a fundamental element in the design of digital communication and storage systems. One can no longer claim to be a communication or computer system engineer without knowing something about coding.

As we look towards the future, it seems clear that parallel and/or serial concatenated coding and iterative decoding (i.e., turbo coding) are destined to replace convolutional codes in many applications that require moderate BER's and can tolerate significant decoding delay. This is another illustration of the importance of minimizing the number of nearest neighbor codewords in applications requiring only moderate BER's. More generally, turbo coding has pointed the way towards more creative uses of code concatenation as a means

of achieving exceptional performance with moderate decoding complexity. The need for almost error-free performance in many scientific applications and high-performance commercial systems, such as CD's and computer memories, seems certain to continue to spur interest in extremely powerful code designs with very large minimum distance, such as algebraic-geometry codes and multilevel construction techniques. And further advances in signal-processing technology are sure to speed the conversion of almost all decoding machines to the use of soft decisions. Finally, coding will continue to find use on increasingly difficult channels as demands for more reliable communications and data storage accelerate. The technical challenges inherent in designing appropriate codes for the many anticipated new mobile communication services, which must operate in severe multipath and fading conditions, and high-density data storage applications, with increasingly difficult read/write signal processing requirements, promise to keep coding theory an active research field well into the next century.

ACKNOWLEDGMENT

The authors wish to thank Dr. Lance C. Perez, Dr. G. David Forney, Jr., Dr. Shu Lin, Dr. Robert H. Morelos-Zaragoza, and Dr. Oscar Y. Takeshita as well as Hermano A. Cabral for valuable comments that contributed to the preparation of this paper.

REFERENCES

- [1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 1948.
- [2] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*. New York: Wiley, 1965.
- [3] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, pp. 147–150, 1950.
- [4] I. M. Jacobs, "Practical applications of coding," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 305–310, May 1974.
- [5] G. D. Forney, Jr., and L. F. Wei, "Multidimensional constellations—Part I: Introduction, figures of merit, and generalized cross constellations," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 877–892, Aug. 1989.
- [6] J. L. Massey, "Deep-space communication and coding: A marriage made in heaven," in *Lecture Notes on Control and Information Sciences* 82, J. Hagenauer, Ed., Bonn, Germany: Springer-Verlag, 1992.
- [7] R. R. Green, "A serial orthogonal decoder," in *Jet Propulsion Laboratory Space Programs Summary*, vol. IV, no. 37–39, June 1966, pp. 247–251.
- [8] J. M. Wozencraft and B. Reiffen, *Sequential Decoding*. Cambridge, MA: MIT Press, 1961.
- [9] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–74, Apr. 1963.
- [10] S. Lin and H. Lyne, "Some results on binary convolutional code generators," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 134–139, Jan. 1967.
- [11] J. L. Massey and D. J. Costello, Jr., "Nonsystematic convolutional codes for sequential decoding in space applications," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 806–813, Oct. 1971.
- [12] J. W. Layland and W. A. Lushbaugh, "A flexible high-speed sequential decoder for deep space channels," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 813–820, Oct. 1971.
- [13] G. D. Forney, Jr., and E. K. Bower, "A high-speed sequential decoder: Prototype design and test," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 821–835, Oct. 1971.
- [14] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.
- [15] A. J. Viterbi, J. K. Wolf, E. Zehavi, and R. Padovani, "A pragmatic approach to trellis-coded modulation," *IEEE Commun. Mag.*, vol. 27, pp. 11–19, July 1989.
- [16] G. Fettweis and H. Meyr, "High-speed parallel viterbi decoding: Algorithm and VLSI-architecture," *IEEE Commun. Mag.*, vol. 29, pp. 46–55, May 1991.
- [17] W. W. Wu, E. F. Miller, W. L. Pritchard, and R. L. Pickholtz, "Mobile satellite communications," *Proc. IEEE*, vol. 82, pp. 1431–1448, Sept. 1994.
- [18] G. D. Forney, Jr., *Concatenated Codes*. Cambridge, MA: MIT Press, 1966.
- [19] Consultative Committee for Space Data Systems, "Recommendations for space data standard: Telemetry channel coding," Blue Book Issue 2, CCSDS 101.0-B2, Jan. 1987.
- [20] R. J. McEliece and L. Swanson, "Reed–Solomon codes and the exploration of the solar system," in *Reed–Solomon Codes and Their Applications*, S. B. Wicker and V. K. Bhargava, Eds. Piscataway, NJ: IEEE Press, 1994, pp. 25–40.
- [21] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968. (Revised edition, Laguna Hills, CA: Aegean Park, 1984.)
- [22] J. L. Massey, "Shift register synthesis and BCH decoding," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 122–127, Jan. 1969.
- [23] J. Hagenauer, E. Offer, and L. Papke, "Matching Citerbi decoders and Reed–Solomon decoders in concatenated systems," in *Reed–Solomon Codes and Their Applications*, S. B. Wicker and V. K. Bhargava, Eds. Piscataway, NJ: IEEE Press, 1994, pp. 242–271.
- [24] O. M. Collins, "The subtleties and intricacies of building a constraint length 15 convolutional decoder," *IEEE Trans. Commun.*, vol. 40, pp. 1810–1819, Dec. 1992.
- [25] S. B. Wicker, "Deep space applications," in *CRC Handbook on Coding Theory*, V. Pless, W. C. Huffman, and R. Brualdi, Eds. Boca Raton, FL: CRC, ch. 25, to be published in 1998.
- [26] E. Paaske, "Improved decoding for a concatenated coding system recommended by CCSDS," *IEEE Trans. Commun.*, vol. COM-38, pp. 1138–1144, Aug. 1990.
- [27] O. M. Collins and M. Hizlan, "Determinate state convolutional codes," *IEEE Trans. Commun.*, vol. 41, pp. 1785–1794, Dec. 1993.
- [28] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. 1989 IEEE Global Communications Conference* (Dallas, TX, Nov. 1989), pp. 47.1.1–47.1.7.
- [29] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. 1993 IEEE Int. Communications Conf.* (Geneva, Switzerland, May 1993), pp. 1064–1070.
- [30] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 409–428, Mar. 1996.
- [31] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.
- [32] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1996.
- [33] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, pp. 909–926, May 1998.
- [34] G. D. Forney, Jr., "Coding and its application in space communications," *IEEE Spectrum*, vol. 7, pp. 47–58, June 1970.
- [35] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 55–67, Jan. 1982.
- [36] ———, "Trellis-coded modulation with redundant signal sets—Part I: Introduction," *IEEE Commun. Mag.*, vol. 25, pp. 5–11, Feb. 1987.
- [37] ———, "Trellis-coded modulation with redundant signal sets—Part II: State of the art," *IEEE Commun. Mag.*, vol. 25, pp. 12–21, Feb. 1987.
- [38] G. D. Forney, Jr., "Coded modulation for band-limited channels," *IEEE Inform. Theory Soc. Newslet.*, vol. 40, pp. 1–7, Dec. 1990.
- [39] L. F. Wei, "Rotationally invariant convolutional channel coding with expanded signal space. Part II: Nonlinear codes," *IEEE J. Select. Areas Commun.*, vol. SAC-2, pp. 672–686, Sept. 1984.
- [40] ———, "Trellis-coded modulation with multidimensional constellations," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 483–501, July 1987.
- [41] M. V. Eyuboglu, G. D. Forney, Jr., P. Dong, and G. Long, "Advanced modulation techniques for V. fast," *Euro. Trans. Telecommun.*, vol. 4, pp. 243–256, May–June 1993.
- [42] G. Lang and F. Longstaff, "A Leech lattice modem," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 968–973, Aug. 1989.
- [43] British Telecom, "Code choice for V. fast," Contribution D0, CCITT Study Group 14, Geneva, Switzerland, Aug. 1993.
- [44] A. R. Calderbank and L. H. Ozarow, "Nonequiprobable signaling on the Gaussian channel," *IEEE Trans. Inform. Theory*, vol. 36, pp. 726–740, July 1990.

- [45] T. R. N. Rao and E. Fujiwara, *Error-Control Coding for Computer Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
- [46] H. Imai, Ed., *Essentials of Error-Control Coding Techniques*. San Diego, CA: Academic, 1990.
- [47] C. L. Chen, "Symbol error-correcting codes for computer memory systems," *IEEE Trans. Comp.*, vol. 41, pp. 252–256, Feb. 1992.
- [48] E. Fujiwara, "Application of coding theory to computer systems," in *Proc. 1997 IEICE General Conf.*, Mar. 1997, vol. TA-3-8, pp. 572–573.
- [49] P. Fire, "A class of multiple-error-correcting binary codes for non-independent errors," Sylvania Rep. RSL-E-2, Sylvania Electronics Defense Lab., Reconnaissance Syst. Div., Mountain View, CA, Mar. 1959.
- [50] K. A. S. Immink, "RS codes and the compact disc," in *Reed-Solomon Codes and Their Applications*, S. B. Wicker and V. K. Bhargava, Eds. Piscataway, NJ: IEEE Press, 1994, pp. 41–59.
- [51] H. Imai and Y. Nagasaka, "On decoding methods for double-encoding systems," *Electron. Commun. Japan*, vol. 65-A, pp. 36–44, Dec. 1982.
- [52] M. Hattori, N. Ohya, M. Sasano, K. Sato, and N. Shirota, "Improved general purpose Reed-Solomon erasure encoder/decoder chip," in *Proc. 1993 Picture Coding Symp.*, Mar. 1993.
- [53] E. J. Weldon Jr., "Difference set cyclic codes," *Bell Syst. Tech. J.*, vol. 45, pp. 1045–1055, 1966.
- [54] T. Kuroda, M. Takada, T. Isobe, and O. Yamada, "Transmission scheme of high-capacity FM multiplex broadcasting system," *IEEE Trans. Broadcasting*, vol. 42, pp. 245–250, Sept. 1996.
- [55] O. Yamada, "Development of an error-correction method for data packet multiplexed with TV signals," *IEEE Trans. Commun.*, vol. COM-35, pp. 21–31, Jan. 1987.
- [56] J. L. Massey, *Threshold Decoding*. Cambridge, MA: MIT Press, 1963.
- [57] M. Takada and T. Kuroda, "Threshold decoding algorithm using trellis diagram for majority logic decodable codes," in *Proc. 1996 Int. Symp. Information Theory and Its Applications* (Victoria, BC, Canada, Sept. 1996), vol. 1, pp. 397–400.
- [58] K. Yamaguchi, H. Iizuka, E. Nomura, and H. Imai, "Variable threshold soft decision decoding," *IEICE Trans.*, vol. 71-A, pp. 1607–1614, Aug. 1988 (in Japanese). English version available in *Electron. Commun. Japan*, vol. 72, pp. 65–74, Sept. 1989.
- [59] ISO-IEC, "Coding of moving pictures and associated audio up to about 1.5 Mbit/s," ISO-ICE, Std. CD 11172-3, Part 3—Audio.
- [60] J. Hagenauer, "Rate-compatible punctured convolutional codes and their applications," *IEEE Trans. Commun.*, vol. 36, pp. 389–400, Apr. 1988.
- [61] G. Plenge, "A new sound broadcasting system," *EBU Tech. Rev.*, vol. 246, pp. 87–112, Apr. 1991.
- [62] C. Weck, "Unequal error protection for digital sound broadcasting—Principle and performance," presented at the 94th AES Conv., preprint 3459, Berlin, Germany, Mar. 1993.
- [63] H. Imai and S. Hirakawa, "A new multilevel coding method using error-correcting codes," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 371–377, May 1977.
- [64] R. H. Morelos-Zaragoza, M. P. C. Fossorier, S. Lin, and H. Imai, "Multilevel block coded modulation with unequal error protection," in *Proc. 1997 IEEE Int. Symp. Information Theory* (Ulm, Germany, July 1997), p. 441.
- [65] J. Hagenauer, "Zur Kanalkapazität bei Nachrichtenkanälen mit Fading und gebündelten Fehlern," *AEU, Electron. Commun.*, vol. 34, pp. 229–237, 1980.
- [66] Qualcomm Inc., "An overview of the application of code division multiple access to digital cellular systems and personal cellular networks," Qualcomm, Inc., San Diego, CA, May 1992.
- [67] J. Proakis, *Digital Communications*. New York: McGraw-Hill, 1989.
- [68] J. Hagenauer, "Source controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, pp. 2449–2457, Sept. 1995.
- [69] C. Erben, T. Hindelang, and W. Xu, "Quality enhancement of coded and corrupted speeches in GSM mobile systems using residual redundancy," in *Proc. ICASSP97* (Munich, Germany, Apr. 1997), vol. 1, pp. 259–262.
- [70] W. Xu, J. Hagenauer, and J. Hollmann, "Joint source-channel decoding using the residual redundancy in compressed images," in *Proc. 1996 Int. Conf. Communications* (Dallas, TX, June 1996), pp. 142–148.
- [71] J. Hagenauer, "The turbo principle: Tutorial introduction and state of the art," in *Proc. Int. Symp. Turbo Codes* (Brest, France, Sept. 1997), pp. 1–11.
- [72] F. Burkert, G. Caire, J. Hagenauer, T. Hindelang, and G. Lechner, "Turbo decoding with unequal error protection applied to GSM speech coding," in *Proc. 1996 IEEE Global Communications Conf.* (London, U.K., Nov. 1996), pp. 2044–2048.
- [73] R. Herzog, A. Schmidbauer, and J. Hagenauer, "Iterative decoding and despreading improves CDMA-systems using M -ary orthogonal modulation and FEC," in *Proc. 1997 IEEE Int. Conf. Communications* (Montreal, Que., Canada, June 1997).
- [74] F. Halsall, *Data Communications, Computer Networks, and Open Systems*, 4th ed. Wokingham, U.K.: Addison-Wesley, 1995.
- [75] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ: Prentice Hall, 1995.
- [76] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice Hall, 1987.
- [77] G. Castagnoli, J. Ganz, and P. Graber, "Optimum cyclic redundancy-check codes with 16-bit redundancy," *IEEE Trans. Commun.*, vol. 38, pp. 111–114, Jan. 1990.
- [78] P. Merkey and E. C. Posner, "Optimum cyclic redundancy codes for noisy channels," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 865–867, Nov. 1984.
- [79] K. A. Witzke and C. Leung, "A comparison of some error detecting CRC code standards," *IEEE Trans. Commun.*, vol. COM-33, pp. 996–998, Sept. 1985.
- [80] R. J. Benice and A. H. Frey, "An analysis of retransmission systems," *IEEE Trans. Commun. Technol.*, vol. COM-12, pp. 135–145, Dec. 1964.
- [81] S. Lin, D. J. Costello Jr., and M. J. Miller, "Automatic-repeat-request error control schemes," *IEEE Commun. Mag.*, vol. 22, pp. 5–17, Dec. 1984.
- [82] R. L. Freeman, *Practical Data Communications*. New York: Wiley, 1995.
- [83] D. E. Comer, *Internetworking with TCP/IP*, vol. I, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1995.
- [84] E. Murphy, S. Hayes, and M. Enders, *TCP/IP, Tutorial and Technical Overview*, 5th ed. Upper Saddle River, NJ: Prentice-Hall, 1995.
- [85] J. M. Wozencraft and M. Horstein, "Digitalised communication over two-way channels," presented at the Fourth London Symp. Information Theory, London, U.K., Sept. 1960.
- [86] ———, "Coding for two-way channels," Tech. Rep. 383, Res. Lab. Electron., MIT, Cambridge, MA, Jan. 1961.
- [87] S. B. Wicker, "Reed-Solomon error control coding for data transmission over Rayleigh fading channels with feedback," *IEEE Trans. Veh. Technol.*, vol. 41, pp. 124–133, May 1992.
- [88] A. Drukarev and D. J. Costello Jr., "Hybrid ARQ error control using sequential decoding," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 521–535, July 1983.
- [89] H. Yamamoto and K. Itoh, "Viterbi decoding algorithm for convolutional codes with repeat request," *IEEE Trans. Inform. Theory*, vol. IT-26, pp. 540–547, Sept. 1980.
- [90] P. Sindhu, "Retransmission error control with memory," *IEEE Trans. Commun.*, vol. COM-25, pp. 473–479, May 1977.
- [91] 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.
- [92] H. Krishna and S. Morgera, "A new error control scheme for hybrid-ARQ systems," *IEEE Trans. Commun.*, vol. COM-35, pp. 981–990, Oct. 1987.
- [93] S. Morgera and V. Oduol, "Soft decision decoding applied to the generalized type-II hybrid-ARQ scheme," *IEEE Trans. Commun.*, vol. 37, pp. 393–396, Apr. 1989.
- [94] S. Lin and P. S. Yu, "A hybrid-ARQ scheme with parity retransmission for error control of satellite channels," *IEEE Trans. Commun.*, vol. COM-30, pp. 1701–1719, July 1982.
- [95] M. B. Pursley and S. D. Sandberg, "Incremental redundancy transmission for meteor burst communications," *IEEE Trans. Commun.*, vol. 39, pp. 689–702, May 1991.
- [96] S. B. Wicker and M. Bartz, "Type-II hybrid-ARQ protocols using punctured MDS codes," *IEEE Trans. Commun.*, vol. 42, pp. 1431–1440, Apr. 1994.
- [97] ———, "The design and implementation of type-I and type-II hybrid-ARQ protocols based on first-order Reed-Muller codes," *IEEE Trans. Commun.*, vol. 42, pp. 979–987, Mar. 1994.
- [98] G. Benelli, "An ARQ scheme with memory and soft error detectors," *IEEE Trans. Commun.*, vol. 33, pp. 285–288, Mar. 1985.
- [99] ———, "An ARQ scheme with memory and integrated modulation," *IEEE Trans. Commun.*, vol. COM-35, pp. 689–697, July 1987.
- [100] J. Metzner, "Improvements in block-retransmission schemes," *IEEE Trans. Commun.*, vol. COM-27, pp. 524–532, Feb. 1979.
- [101] J. Metzner and D. Chang, "Efficient selective-repeat ARQ strategies for very noisy and fluctuating channels," *IEEE Trans. Commun.*, vol. COM-33, pp. 409–416, May 1985.
- [102] B. A. Harvey and S. B. Wicker, "Packet combining systems based on the Viterbi decoder," *IEEE Trans. Commun.*, vol. 42, pp. 1544–1557, Apr. 1994.