Code Combining—A Maximum-Likelihood Decoding Approach for Combining an Arbitrary Number of Noisy Packets

DAVID CHASE, SENIOR MEMBER, IEEE

Abstract—**It is well known that if the data rate is chosen below the available channel capacity, error-free communication is possible. Furthermore, numerous practical error-correction coding techniques exist which can be chosen to meet the user's reliability constraints. However, a basic problem in designing a reliable digital communication system is still the choice of the actual code rate. While the popular rate-1/2 code rate is a reasonable, but not optimum, choice for additive Gaussian noise channels, its selection is far from optimum for channels where a high percentage of the transmitted bits are destroyed by interference. Code combining represents a technique of matching the code rate to the prevailing channel conditions. Information is transmitted in packet formats which are encoded with a relatively high-rate code, e.g., rate 1/2, which can be repeated to obtain reliable communications when the redundancy in a rate-1/2 code is not sufficient to overcome the channel interference. The receiver combines noisy packets (*code combining*) to obtain a packet with a code rate which is low enough such that reliable communication is possible even for channels with extremely high error rates. By combining the minimum number of packets needed to overcome the channel conditions, the receiver optimizes the code rate and minimizes the delay required to decode a given packet. Thus, the receiver adapts to the actual jammer-to-signal (*J/S*) ratio which is critical when the level of interference *J* is not known *a priori.***

I. Introduction

For many communications applications, the concept of determining the channel's data capacity and selecting an appropriate modulation and coding approach, represents a sound approach to arriving at a respectable system. Unfortunately, there are some channels where the initial steps of modeling the channel and determining the data rate that it can support are, at best, an educated guess. Some examples are channels with nonstationary noise, such as the wide-band HF channel, as well as rapidly time-varying channels, such as a meteor-burst channel. Even the relatively benign satellite and LOS channels can fall into this category when a hostile and unknown interfering signal is also present. This paper presents a practical and effective communication approach, code combining, for overcoming the problem of obtaining reliable communications when the actual channel capacity is unknown.

II. Elements of Code Combining

In this section the elements of code combining are described. Briefly, code combining represents a technique for combining the minimum number of repeated packets encoded with a code of rate R to obtain a lower rate, and thus more powerful, error-correcting code, capable of allowing communications when channel error rates are less than 50 percent. Two critical features of code combining are as follows:

1) maximum-likelihood decoding, rather than bounded minimum distance (algebraic) decoding, of L noisy packets as a single low-rate code of rate R/L;

2) weighting of each packet by an estimate of its reliability (soft decision on packets).

Packets can be combined in any order, so that even if several packets are lost, or equivalently, given a reliability weight of 0, decoding is still possible.

Paper approved by the Editor for Communication Theory of the IEEE Communications Society for publication after presentation at MILCOM, Los Angeles, CA, October 1984. Manuscript received April 3, 1984; revised August 14, 1984.

The author is with CNR, Inc., Needham, MA 02194.

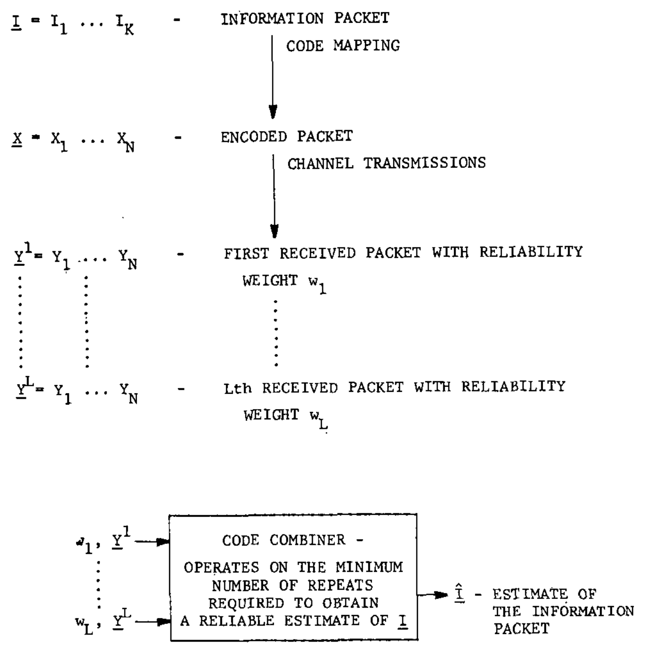

$\underline{I} = I_1 \ldots I_K$ - INFORMATION PACKET
CODE MAPPING
$\underline{X} = X_1 \ldots X_N$ - ENCODED PACKET
CHANNEL TRANSMISSIONS
$\underline{Y}^1 = Y_1 \ldots Y_N$ - FIRST RECEIVED PACKET WITH RELIABILITY WEIGHT w_1
$\underline{Y}^L = Y_1 \ldots Y_N$ - Lth RECEIVED PACKET WITH RELIABILITY WEIGHT w_L
$w_1, \underline{Y}^1$
$w_L, \underline{Y}^L$
CODE COMBINER - OPERATES ON THE MINIMUM NUMBER OF REPEATS REQUIRED TO OBTAIN A RELIABLE ESTIMATE OF $\underline{I}$
$\hat{\underline{I}}$ - ESTIMATE OF THE INFORMATION PACKET

Fig. 1. Sequence of mappings used in code combining.

Fig. 1 illustrates the sequence of mappings used in code combining. We start with an information packet I composed of K bits, which is mapped into an encoded packet with a code rate of R. While a code rate $R = 1/2$ is a typical choice, the actual information packet may, depending on the application, contain additional control overhead. For example, if a convolutional code is used, a truncation tail is required. For both block and convolutional codes, a parity check sequence (error-detection code) is treated as part of the information packet, so that the decoder can determine when to stop combining packets. Thus, the information packet I is actually an error-detection code. To obtain an estimate of the individual packet reliability, additional overhead bits may also be added. We refer to the decoder as a code combiner since it combines successive received packets until the code rate is low enough to provide a successful decode. For the example shown in Fig. 1, a code rate of R/L is assumed to be sufficient for successful decoding.

A possible packet format may be an information packet I

of 2000 bits which is rate-1/2 encoded to obtain a 4000 bit packet. The information packet would contain a parity check sequence on the order of 24 bits for error detection (undetected error rate $<2^{-24}$) so that the receiver can determine when to stop code combining. Depending on the applications the true information bits may be further reduced below 1976 bits as further discussed in Section II-B. However, the difference in rate between the true information and the information packet $\boldsymbol{I}$ is typically quite small.

The code combiner treats the received packets $\boldsymbol{Y}^1, \boldsymbol{Y}^2, \cdots, \boldsymbol{Y}^L$ as a code of rate R/L and attempts to pick the most probable information packet. Each packet is assumed to have a reliability weight of w_i. The decoder may be a rate-R/L soft or hard decoder, but will also have the capability of weighting the reliability of each received packet. It is important to note that code combining is designed to work in a very noisy (jamming) environment, where conventional diversity combining concepts can easily break down. Thus, the diversity combining concept of "adding up" the symbols in the received packets and decoding with a code rate of R is effective for simple fading channels but not necessarily effective in the presence of jamming. Note, with diversity combining, a single error due to a large undetected jammer can cause an entire sum of L received symbols to be incorrect, while with code combining, a single error in L symbols can typically be corrected by the more powerful rate R/L code. Code combining represents a technique for combining entire packets, which represents an added dimension to diversity concepts which are limited to combining just individual symbols.

The conventional ARQ approach (see the Applications Section, III-D) of repeating a packet until a good one is received can be effective for a burst-error channel [6], but is ineffective when channel conditions are such that all packets contain errors. By combining these noisy packets in an optimum manner, significant data throughput is obtained even when the conventional approach of just repeating packets fails. References [7] and [8] suggest techniques for combining pairs of packets to improve performance, but again the performance is limited by error-correction capabilities obtained from only two packets. The technique of [8] has the interesting property that the two packets combined do not represent the same codeword. An information section is sent first and a parity section is sent only if the information section is not received correctly. However, the performance is limited to that possible for a rate-1/2 code. Reference [9] uses a similar scheme to that discussed in [8], but also suggests symbol-by-symbol diversity combining before rate-1/2 decoding. As noted above, these diversity concepts may be ineffective in a jamming environment and cannot benefit from the packet reliability weighting suggested in this paper. Code combining can allow communication as long as the channel capacity is finite and is not limited by the performance of rate-1/2 codes (error rate must be below 12 percent even with maximum-likelihood decoding—see Section II-D), or the performance of algebraic decoders (error rate must be below 25 percent regardless of the code rate—see Section II-A).

A. Code Selection

We desire a coding technique which can operate in a very high error environment as well as under benign channel conditions. For this reason, a maximum-likelihood decoder, rather than an algebraic decoder, is critical. A maximum-likelihood decoder makes the best possible estimate, while algebraic decoders are only effective when the noise is below some threshold which allows the decoder to operate successfully when the number of errors is less than one-half the code's minimum distance d.

We begin our code selection with a maximum-likelihood (Viterbi) decoder and a rate-1/2 constraint length 7 convolutional code [1]. This code is of moderate complexity and, more importantly, retains its effectiveness as it is repeated. Many other codes may also be considered including short block codes. Note that for both block and convolutional codes, the minimum distance for a code repeated L times is simply L times the minimum distance of the basic code.

TABLE I
MINIMUM FREE DISTANCE FOR A REPEATED $K = 7$ CONVOLUTIONAL CODE COMPARED TO UPPER BOUNDS ON OPTIMUM CODES

NUMBER OF REPEATS	R = CODE RATE	d = MINIMUM FREE DISTANCE FOR THE SELECTED CODE	d' = UPPER BOUND ON THE MINIMUM FREE DISTANCE
1	1/2	10	10
2	1/4	20	20
4	1/8	40	41
8	1/16	80	82
16	1/32	160	164
32	1/64	320	329
64	1/128	640	658

The minimum free distance of a convolutional code is upper bounded [2] by

$$d' \leqslant \min_h \frac{2^h}{2^h - 1} \frac{K + h - 1}{2R} \tag{1}$$

where R is the code rate, K is the constraint length, and h is a parameter which is chosen to optimize the bound. For a constraint length $K = 7$, the tightest bound is obtained when $h = 3$. Table I compares the minimum distance of the repeated $K = 7$ convolutional code and this upper bound on the best $K = 7$ convolutional codes. For $R = 1/2$ and $1/4$ the selected code is optimum in the sense that no convolutional code can be found with a larger minimum distance. For rates below 1/4, the difference between d and d' is small enough that even if codes with distance d' can be found (note d' is only an upper bound on minimum distance) the actual performance difference would be negligible.

As noted earlier, one of the key features of code combining is the use of a maximum-likelihood decoder. As a simple example the minimum distance of the repeated Golay code is listed in Table II. Also listed is the ratio of errors that can be corrected compared to the block length of the repeated code for an algebraic bounded distance decoder. The maximum channel error rate that can be corrected is bounded by 16.7 percent for this repeated Golay code. As is shown in Section II-D, channel error rates considerably higher than this can be corrected by using a repeated Golay code with maximum-likelihood decoding.

In general, for a binary symmetric channel (BSC) with an error probability of p we can define the variable

$$U_i = \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1 - p. \end{cases} \tag{2}$$

The number of errors in a codeword of length N is

$$\alpha = \sum_{i=1}^{N} U_i. \tag{3}$$

This random variable has a mean of pN and a variance of $(1 - p)pN$. When normalized by $1/N$ the mean is p and the variance is $(1 - p)p/N$.

For a code capable of correcting e errors the probability

TABLE II
MINIMUM DISTANCE OF A REPEATED GOLAY CODE AND ITS ALGEBRAIC ERROR-CORRECTION CAPABILITIES

NUMBER OF REPEATS	CODE DIMENSIONS (N,K)	MINIMUM DISTANCE d	ALGEBRAIC ERROR-CORRECTION RATIO e/N
1	(24,12)	8	0.125
2	(48,12)	16	0.146
3	(72,12)	24	0.153
4	(96,12)	32	0.156
8	(192,12)	64	0.161
16	(384,12)	128	0.164
32	(768,12)	256	0.165
64	(1536,12)	512	0.166

of error is

$$p(e) = \Pr\,[\alpha > e] = \Pr\left[\frac{1}{N}\sum_{i=1}^{N} U_i > \frac{e}{N}\right]. \tag{4}$$

For large values of N we can take expected values to obtain

$$p(e) = \begin{cases} \to 0 & \text{for } p = E\left[\frac{\alpha}{N}\right] < \frac{e}{N} \\ \to 1 & \text{for } p = E\left[\frac{\alpha}{N}\right] > \frac{e}{N}. \end{cases} \tag{5}$$

Note, the variance of α/N goes to zero for large N.

The above results when applied to the Golay code show that if a bounded minimum distance decoder is used, for a large number of repeats, we have

$$p(e) \to 0 \qquad \text{for } p < \frac{1}{6} \text{ (16.7 percent)}$$

$$p(e) \to 1 \qquad \text{for } p > \frac{1}{6} \text{ (16.7 percent).} \tag{6}$$

Similarly, for the best block codes we have upper and lower bounds on the minimum distance [3], which state that as $R \to 0$

$$\frac{d}{N} \to \frac{1}{2}. \tag{7}$$

Thus, for any bounded distance decoder we must have

$$\frac{e}{N} \leqslant \frac{1}{4} \tag{8}$$

so that

$$p(e) \to 1 \qquad \text{for } p > \frac{1}{4} \text{ (25 percent).} \tag{9}$$

The above arguments show that bounded distance algebraic decoders cannot operate for $p > 25$ percent; however, the positive channel capacity ($C = 0.1887$, at $p = 25$ percent) indicates that error-free communication is still possible. Code combining represents a practical way to communicate over all channels with a positive channel capacity, i.e., for channel error rates with $p < 0.50$.

B. Packet Format

A straightforward technique for implementing code combining is to use a packet format composed of three components illustrated as follows:

$$\text{packet format} = [\text{sync} \mid \text{data} \mid \text{CRC}]. \tag{10}$$

The same packet is transmitted a fixed number of times until feedback is obtained requiring a new packet, or for example, in a broadcast mode, the packet is transmitted for a predetermined time.

The synchronization bits are initially used to detect the presence of a packet and the start of the data bits. After initial synchronization, the synchronization bits may be used to estimate the reliability of successive packets. This information is used by the code combiner to optimize the performance. Another use of the synchronization bits is to distinguish between different packets to prevent false code combining. This property is obtained by simply inverting the sync bits when a new packet is to be repeated. For some applications, the sync bits may be omitted with the coded (redundant) data being used to achieve the required sync functions. In general, for jamming scenarios, only a portion of the sync bits would be placed at the start of the packet, with the remainder being pseudorandomly interleaved over the duration of the packet.

The data bits are composed of a meaningful number of characters and may contain control information needed for network applications. The cyclic redundancy check (CRC) bits form an error-detection code and are used by the maximum-likelihood decoder to determine if the packet has been decoded correctly. Code combining stops when the CRC bits indicate an error-free packet.

The data and the CRC are encoded with a basic code of rate R which is used for code combining successive packets. In Fig. 1 the information packet I represents both the *data* and *CRC* bits. The synchronization bits represent a 1 bit code (assuming sync bits are inverted for each packet change) which can also be code combined over successive packets.

When a packet is first received it is decoded by a rate R maximum-likelihood decoder. The CRC bits are checked after decoding. If errors are still detected, this packet is saved and combined with the next received packet by using a rate-$R/2$ maximum-likelihood decoder. If the CRC bits still do not check, a third packet is combined by using an $R/3$ decoder and so on, until a successful decode is obtained. In practice, the structure of a maximum-likelihood rate R/L decoder is quite similar for all L as illustrated in the following section.

C. Reliability Weighting of Packets

The concept of repeating packets to obtain a near optimum low-rate code can offer a significant advantage over using a fixed low-rate code, even if one can choose the appropriate code rate *a priori*.

This is particularly true for applications where repeated packets have significantly different reliability. For example, repeated packets may be transmitted over different frequency subbands with the channel error rate varying significantly with the subband chosen. Similarly, a time-varying jammer can result in large variations in the reliability of successive packets.

For these applications, packets can be weighted by their reliability so as to optimize the code combining performance. In fact, 'bad packets"[1] can be completely ignored. Typically, a fixed low-rate code accepts the portion of data which is jammed and relies on interleaving to randomize the jammed bits. Thus, code combining offers the additional feature of having a reliability measure on each packet which is unavaila-

[1] Channel error rates approaching 50 percent.

ble when a fixed low-rate coding approach is used. This reliability measure may be viewed as a soft decision on a packet as opposed to the more conventional soft decisions used on a bit-by-bit basis.

An example of how packets may be weighted can be obtained by considering the transmission of L packets of N bits, each over a binary symmetric channel (BSC) with probability of bit error for each packet given by p_i for $i = 1, 2, \cdots, L$, A maximum-likelihood decoder will select the codeword (packet) m which maximizes the conditional probability between the received sequence Y and the repeated packet denoted as X_m. This function can be written as

$$\max_m \left\{ p[Y|X_m] = \prod_{i=1}^{L} (1-p_i)^{N-d_{mi}} p_i^{d_{mi}} \right\} \tag{11}$$

where d_{mi} is the number of bit disagreements for the ith packet. Taking the natural logs of both sides yields

$$\max_m \{\ln p[Y|X_m]\} = \max_m \left[\sum_{i=1}^{L} N \ln(1-p_i) - d_{mi} \ln\left(\frac{1-p_i}{p_i}\right) \right]. \tag{12}$$

Finally, by dropping the $N \ln(1 - p_i)$ term which is not a function of m and omitting the negative sign by taking a min rather than max, we obtain

$$\max_m \{\ln p[Y|X_m]\} \propto \min_m \sum_{i=1}^{L} d_{mi} \ln\left(\frac{1-p_i}{p_i}\right). \tag{13}$$

Thus, for the above example a maximum-likelihood decoder picks the packet with the minimum number of disagreements weighted by the reliability factor.

$$w_i = \ln\left(\frac{1-p_i}{p_i}\right). \tag{14}$$

For this example, the code combiner operates with a hard decision on individual bits (BSC). However, a soft decision on each individual packet is used.

To apply this reliability factor to a packet, an estimate of the channel error rate p_i for each packet must be obtained. One approach is simply to count the errors in the known synchronization sequence inserted in each packet. As noted earlier, a portion of the sync sequence may be pseudorandomly interleaved within the packet for this purpose. Another approach is to record the number of errors a decoder would correct when operating on an uncombined packet. Simply comparing the bit-by-bit estimate obtained from code combining the previous packets to the newest packet can be quite effective even before the CRC checks. Signal-to-noise estimations obtained from the demodulator may also be used to obtain an estimate of p_i.

An alternate way to write (13) is

$$\min_m \sum_{j=1}^{L} \ln\left(\frac{1-p_j}{p_j}\right) \sum_{i=1}^{N} Y_i^j \oplus X_{mi}. \tag{15}$$

If we assume all repeats have the same reliability, we can drop $\ln[(1 - p_j)/p_j]$ and interchange the order of summation to give

$$\min_m \sum_{i=1}^{N} \sum_{j=1}^{L} Y_i^j \oplus X_{mi}. \tag{16}$$

The inner summation can be any value over the range $0, 1, \cdots, L$, depending on the reliability of the ith bit. Thus, the code combiner for a BSC is actually using a soft decision on each symbol, followed by a nonrepeated (rate-1/2) maximum-likelihood decoder. The factor $\ln(1 - p_j/p_j)$ can still be inserted inside the double summation in (16) to represent the more general case. Thus, the optimum code combiner for a BSC can also be viewed as a rate-1/2 maximum-likelihood soft decoder.

For a Gaussian channel, the code combiner structure is given in terms of the received sequence V as

$$\max_m \left\{ p(V|X_m) = \prod_{j=1}^{L} \prod_{i=1}^{N} \frac{e^{-(v_i - x_{mi})^2/2\sigma_j^2}}{\sqrt{2\pi\sigma_j^2}} \right\} \tag{17}$$

where $x_{mi} = \pm 1$, $v_i = x_{mi} + n_i$, and n_i is a Gaussian variable of variance σ_j^2. Taking natural logs and dropping terms which are not a function of m we obtain

$$\max_m \{p(V|X_m)\} \propto \max_m \sum_{j=1}^{L} \frac{1}{\sigma_j^2} \sum_{i=1}^{N} v_i x_{mi}. \tag{18}$$

The above expression can also be written in a form similar to (13) as

$$\min_m \sum_{j=1}^{L} \frac{1}{\sigma_j^2} \sum_{i=1}^{N} \alpha_i (Y_i^j \oplus X_{mi}) \tag{19}$$

where $\alpha_i = |v_i|$ is the *symbol* soft decision. The weighting function $1/\sigma_j^2$ is the soft decision function for each repeated packet. In the presence of jamming (see Section III-B) this structure must be modified so that jammed symbols are erased and not included in the correlation function given in (19).

D. Performance Examples

In this section three rate-1/2 codes are evaluated for use in code combining. The constraint length 7 convolutional code, the constraint length 5 convolutional code, and the (24, 12) Golay code represent attractive candidates. Maximum-likelihood decoders can be readily implemented for all three codes. A binary symmetric channel with a *fixed* error rate per packet repetition is assumed, and thus successive packets would not be weighted.

The channel capacity limit C is computed from the channel error probability p as

$$C = 1 - p \log_2 p - (1-p) \log_2 (1-p). \tag{20}$$

This is related to the number of rate-1/2 packet transmissions L as

$$L \geqslant \frac{1}{2C} \tag{21}$$

for the ideal case when $R = C$.

Thus, for example, when $p = 0.20$, the overall code rate

$$R \leqslant C = 0.2781 \tag{22}$$

or the number of rate-1/2 packets must satisfy

$$L \geqslant \frac{1}{2C} = 1.7981. \tag{23}$$

For a decoded error rate of 10^{-4}, and $p = 0.2$, Fig. 2 illustrates that about five packets are required for a $K = 7$ code, about six packets for a $K = 5$ code, and about seven packets for the (24, 12) code. These curves were obtained by actual simulations, but union bounds may be used to provide adequate estimates below the 10^{-4} region.

The important point to note is that for all $p < 0.5$, error-free communication is possible for all three codes, providing the appropriate number of packet repeats are transmitted.

The relationship between the required code rate and the channel error rate can also be seen by plotting the decoded error rate as a function of the channel error rate. Fig. 3 illustrates this type of input–output plot for the $K = 7$ convolutional code used over a BSC for code rates of 1/2, 1/4, ⋯, 1/256. These results can be applied to a wide variety of channels, including jammed channels, providing the channel errors are interleaved so that they appear random to the decoder.

It should be noted that a repeated code shows no coding gain in the required energy-to-noise per information bit (E_b/N_0) for a Gaussian channel, e.g., the rate-1/2 and -1/4 codes are identical when plotted in terms of E_b/N_0. This is also true when comparing spread and nonspread systems in a nonjammed environment. While the advantage of spread-spectrum systems, or more generally systems employing low-rate codes, is well known in a jamming environment, code combining has the added advantage of being adaptive to the actual jamming threat. When communication is marginal, the received E_b/N_0 can be increased by simply combining more packets. These features are further emphasized in the Applications section which follows.

III. Applications

A wide variety of applications of code combining are discussed in this section. Some applications are based on the need for a variable rate error-correction code, for which code combining represents a specific approach, while other applications are based on the fact that code combining minimizes the decoding time needed to receive a reliable packet. In an effort to cover a wide range of applications much of the details needed to actually implement a system are intentionally omitted.

A. Spread-Spectrum Systems

For a fairly wide class of spread-spectrum systems [4] the overall signal-to-noise ratio $E_b/(N_0)_{eq}$ is given in terms of the received signal power S, jammer power J, noise power per Hz N_0, data rate D, and spread-spectrum bandwidth W_s, as

$$\frac{E_b}{(N_0)_{eq}} = \frac{S}{DN_0 + J\left(\frac{D}{W_s}\right)}. \tag{24}$$

The factor W_s/D, called the processing gain, represents the attenuation of the jammer. For a given threat (value of J/S) the data rate may be chosen low enough so that an acceptable value of $E_b/(N_0)_{eq}$ is obtained. Unfortunately, in an actual communication scenario, the jammer may be unknown and could easily exceed the value of J/S for which the system was designed. An effective way of combating this uncooperative jammer is to use a lower rate error-correcting

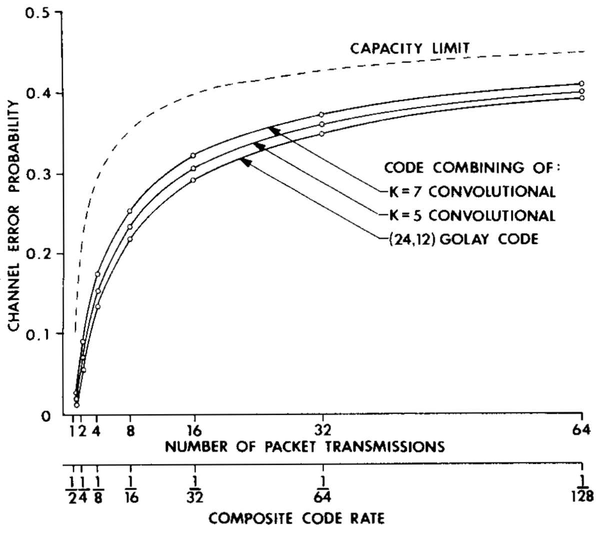


Fig. 2. Comparison of code combining and capacity limits.

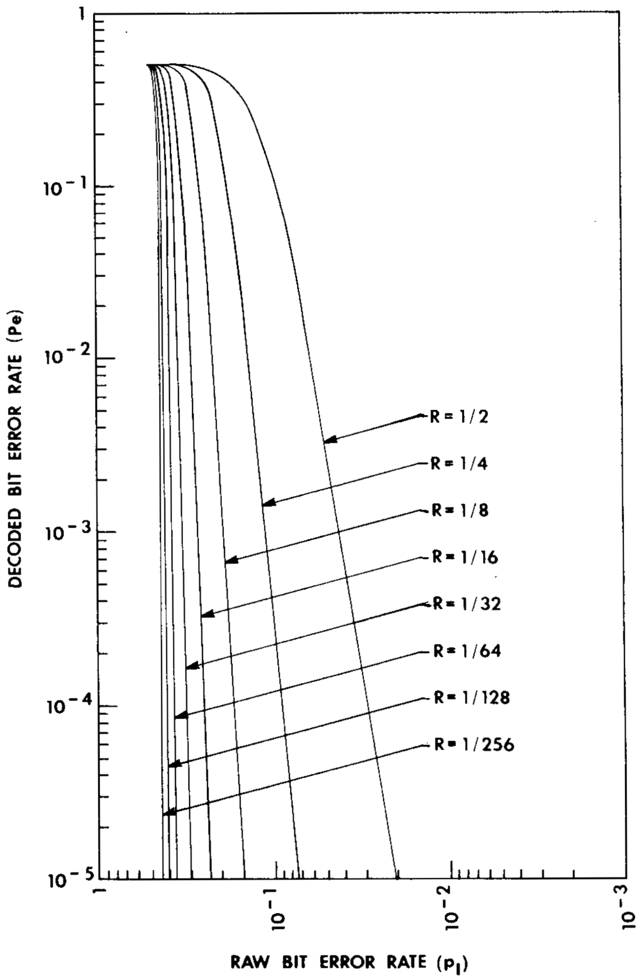


Fig. 3. Simulated performance of $K = 7$ convolutional codes with hard decoding and independent bit errors.

code, which reduces the throughput data rate D, and thus increases the processing gain.

Code combining represents an efficient way of varying the throughput data rate D to adaptively match the actual jammer threat.

B. Impact of Low-Rate Codes on Channel Capacity with Jamming

While channel capacity results are generally not achievable in practice, the results provide insight into the selection of effective codes. As an example, for the additive Gaussian channel without interference, the restriction to binary codes yields no loss in the ultimate performance which is given by $E_b/N_0 \rightarrow -1.6$ dB. This is true when soft decision (channel measurement) decoding information is used in the decoding procedure and the code rate $R \rightarrow 0$. The popular rate-1/2 code has a capacity limit for E_b/N_0 of 0.2 dB; thus for optimum systems a loss in predicted performance of 1.8 dB results if a system is restricted to rate-1/2 binary codes. An additional loss of approximately 2 dB is predicted if binary decoding is used in place of soft decision decoding.

In [5], the additive Gaussian noise model is generalized to include interference. The interference is modeled by a single parameter δ, which represents the probability that the interference coincides with a code symbol. Furthermore, it is assumed that the code symbol is completely destroyed by the interference. Thus, the models evaluated are worst-case models.

The importance of choosing low-rate codes when operating in the presence of interference is illustrated for the Gaussian channel with interference by noting that when $R \rightarrow 0$, with $\delta = 50$ percent, the minimum value of E_b/N_0 needs to be increased to 4.4 dB from the minimum value of -1.6 dB occurring at $\delta = 0$ percent. Furthermore, if a code rate greater than 1/2 is utilized, the results in [5] indicate that error-free communication is impossible for $\delta = 50$ percent, regardless of the available E_b/N_0. The need for low-rate codes is even more pronounced when binary decoding is used. Now for $\delta = 0.5$, the minimum possible value of E_b/N_0 obtained as $R \rightarrow 0$ is 6.4 dB rather than the 4.4 dB value quoted above for the case when soft decision decoding is used. Furthermore, with binary decoding and $\delta = 0.5$, the code rate must be below 0.19 in order to guarantee error-free communications even for arbitrarily high values of E_b/N_0. It should be noted that when no interference is present, $\delta = 0$, the use of binary decoding only increases the required E_b/N_0 and does not impose any restriction on code rate.

Considering that meaningful soft decision information may be difficult to obtain in the presence of jamming, the importance of having a communication system that can vary the code rate should be clear even from these theoretical results.

An important point to note is that in this case it is necessary to use low-rate codes for reliable communications. Simply increasing the processing gain, as suggested in the spread-spectrum application, is not sufficient for this model which assumes that $J/S \rightarrow \infty$ with a duty factor of δ.

C. Packet Communication Systems

Most packet formats contain at least an error detection code and in some cases an error-correction code is also employed. When errors are (after error correction) detected in a packet, the packet is discarded. This design philosophy makes sense in applications where a small percentage of the packets are discarded. Alternate routes may be established or the statistics of the noise may be such that a repeated transmission over the same route is likely to be successful.

For packet radio networks operating in the presence of jamming, one can easily arrive at a situation where almost all packets contain too many errors for correction with a fixed rate code. Rather than ignoring these packets, it is possible to combine multiple repeats of the *same* packet to obtain a more powerful lower rate code which allows the group of the packets to be decoded successfully. Code combining is ideally suited for this packet combining requirement.

D. Two-Way Links (ARQ Code Combing)

Perhaps the simplest application of code combining is in an automatic repeat request (ARQ) two-way communication system. Data are transmitted in blocks, i.e., packets, and repeated if a block is received with errors. A feedback link is used to signify an error-free block (ACK) or a request for a repeat (NACK) when errors are detected. This type of application falls in the class of selective-repeat ARQ strategies [6]. As discussed in Section II, some work related to code combining has been done [7]-[9] in this ARQ application. However, it seems this work has been motivated by relatively quiet (satellite) channels where there is a high probability of receiving a correct packet even without error correction. Code combining, as defined in this paper, is designed to work over very noisy channels, and thus, an arbitrary number of noisy packets may be combined until the code rate is low enough to overcome the interference. For channels which are expected to be very quiet as well as noisy, the basic code rate used by the code combiner would be significantly higher than the rate-1/2 examples mentioned in this paper, e.g., a rate-0.9 may be used.

When code combining is used, successive packet repeats can be combined to obtain a lower, and thus more powerful, error-correcting code. This approach is considerably more effective than waiting for a single reliable packet for noisy channels.

Assuming small transmission delays and fast processing, an odd/even packet format can be used to allow a continuous flow of data based on the feedback information as shown in Fig. 4. Generalization for links with delays (including processing time) of multiple packets is straightforward. The impact of finite buffers on the performance of ARQ code combining is an interesting area which is not treated in this paper. Results are highly dependent on the channel model assumed and the application.

Each informaton packet consists of packet control information (such as packet number), packet data, and a checksum capable of verifying the correctness of the packet. The even/odd packet scheme eliminates the wait for ACK/NACK as follows. The data transmitter will alternate sending two packets referred to as "even" and "odd." After the data receiver processes the "even" packet during the "even" time slot the receiver makes a decision on whether to send an ACK or NACK, which is sent during the "odd" time slot. During the "odd" slot, the receiver processes the "odd" packet and makes a decision whether to ACK or NACK on this. When the transmitter receivers an ACK on one of the packets, it is replaced by a new packet. This scheme can be extended to two-way data transmission by reserving dedicated ACK/NACK time at the end of the information packet.

In the ARQ system, the reliability of the return ACK/NACK must be considered; therefore, the ACK/NACK will be much more heavily coded that the basic packet rate. An unreliable ACK/NACK can be interpreted as a NACK which forces the transmitter to continue to repeat the current packet. This is the correct response if a NACK was transmitted and forces additional repeats if an ACK was actually transmitted. An additional packet can be easily removed after decoding when packets are numbered.

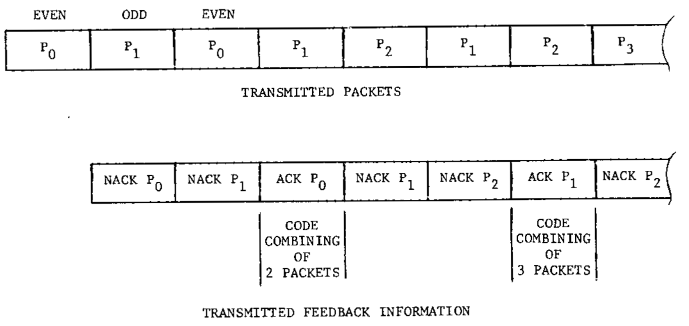


Fig. 4. Odd/even packet scheme with ACK/NACK.

E. Multiple Hop Networks

In this section several examples are given illustrating the importance of code combining in a network. Code combining enables a terminal to relay a message with *minimum delay* which is a critical property required to transfer information in a timely manner. The links are assumed to be half-duplex, i.e., a node can either transmit or receive.

1) Links with Variable Reliability: A sample three-hop example is given in Fig. 5. The transfer of a packet from point A to B is assumed to require four repeats.

If a rate-1/2 constraint length 7 convolutional code is used, four repeats correspond to a rate-1/8 code which yields a decoded error rate of 10^{-5} for a 15 percent channel error rate. Thus, for a packet of 1000 bits and a 15 percent channel error rate, the probability of a successful packet reception with four repeats is above 99 percent. Actually, the dependent errors which result in decoding a convolutional code will make this probability considerably higher than 99 percent. The relationship between the number of repeats used in this example and the channel error is given in Fig. 5.

The link from B to C is assumed highly reliable, and thus one packet is required.

The least reliable link is from C to D and ten packet repeats are assumed to be needed for a reliable reception.

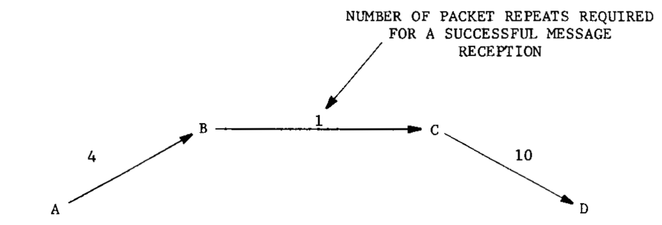
If an ARQ type of feedback is available in this simple network, we see that a packet can be transferred from A to D in a time equal to 15 packet repeats. In fact, even if feedback is unavailable, Fig. 6 illustrates two approaches for transferring a packet from A to D with minimum delay.

As a reference, we note that a transmitter can send each packet ten times, assuming the worst-case delay is known, and achieve a transfer with 30 packets of delay. This *fixed* coding approach is inefficient because the code rate must be chosen low enough to communicate over the *worst-case* link and, furthermore, the transmitter is required to know the worst-case decoding delay. The actual link which is worst-case may not be known.

If it is known that no more than ten repeats are required, code combining may be used as illustrated by the second example in Fig. 6 with each terminal transmitting ten repeats and each receiver relaying as soon as the packet is decoded.

A more attractive approach is illustrated by the last example of Fig. 6 where only the maximum total delay must be known. Since this is a *sum* over several links, this type of an estimate is considerably easier to obtain than on a worst-case estimate for a *single* channel. For this case, each packet is repeated until a predetermined time-out. The use of this concept for broadcasting (flooding) successive messages in a network is attractive since all terminals stop transmitting at the predetermined time-out.

2) Multihop Links with Identical Statistics: While it is intuitively clear that code combining is attractive when successive links have variable reliability, it is instructive to compare fixed rate codes to code combining for multihop links with identical statistics. An actual calculation appears necessary for this application since the true benefits of adaptive code combining for this application are not as intuitive.


NUMBER OF PACKET REPEATS REQUIRED FOR A SUCCESSFUL MESSAGE RECEPTION
A
B
C
D
4
1
10

FOR RATE-1/2 ENCODED PACKETS WITH A CONSTRAINT LENGTH 7 CONVOLUTIONAL CODE AND BINARY MAXIMUM-LIKELIHOOD DECODING, A DECODED ERROR RATE OF 10^{-5} CAN BE ACHIEVED FOR THE FOLLOWING CHANNEL ERROR RATES:

NUMBER OF PACKET REPEATS	CHANNEL ERROR RATES
1	2%
2	7%
3	12%
4	15%
8	23%
10	25%

Fig. 5. Example of a three-hop link with variable reliability.

Results are presented for four- and eight-hop links. The model for the performance of error-correcting is that the decoding time for a single hop is a random variable t of mean u with pdf $f(t)$:

$$f(t) = \frac{1}{u} e^{-t/u}, \qquad t \geqslant 0. \tag{25}$$

For code combining, each relay in a multihop link passes the message on to the next station as soon as it has received the message successfully. Because of this, the decoding time of a multihop link is the sum of decoding times of each hop of the link. For a four-hop link, the decoding time is a random variable $T_4 = t_1 + t_2 + t_3 + t_4$. For an eight-hop link, the decoding time is random variable $T_8 = t_1 + t_2 + \cdots + t_8$.

For a fixed coding scheme, the available time is divided equally among the hops of the link. Successful transmission occurs when all hops achieve transmission in the allotted time.

Fig. 7 show the probability of a successful transmission for four- and eight-hop links when a fixed code rate and adaptive code combining scheme is used.

For a probability of a successful reception of 90 percent, the decoding times in terms of the mean delay per link m are given as:

four-hop adaptive = 7 u

four-hop fixed = 14 u

eight-hop adaptive = 12 u

eight-hop fixed = 35 u.

Note, with code combining an eight-hop message can be received even more quickly than a four-hop message transmitted with a fixed code rate. This result is even more remarkable since the model above is based on all links having the same average mean decoding time of u. Having a variable decoding

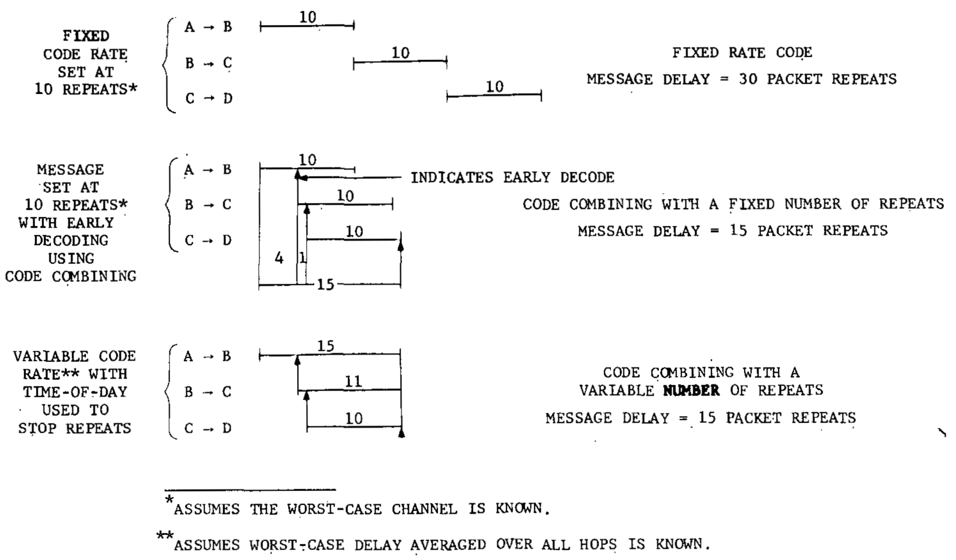

FIXED CODE RATE SET AT 10 REPEATS*
A → B
B → C
C → D
10
FIXED RATE CODE
MESSAGE DELAY = 30 PACKET REPEATS
MESSAGE SET AT 10 REPEATS* WITH EARLY DECODING USING CODE COMBINING
INDICATES EARLY DECODE
4
1
15
CODE COMBINING WITH A FIXED NUMBER OF REPEATS
MESSAGE DELAY = 15 PACKET REPEATS
VARIABLE CODE RATE** WITH TIME-OF-DAY USED TO STOP REPEATS
11
CODE COMBINING WITH A VARIABLE NUMBER OF REPEATS
MESSAGE DELAY = 15 PACKET REPEATS
*ASSUMES THE WORST-CASE CHANNEL IS KNOWN.
**ASSUMES WORST-CASE DELAY AVERAGED OVER ALL HOPS IS KNOWN.

Fig. 6. Comparison of a fixed rate code and code combining for a single message.

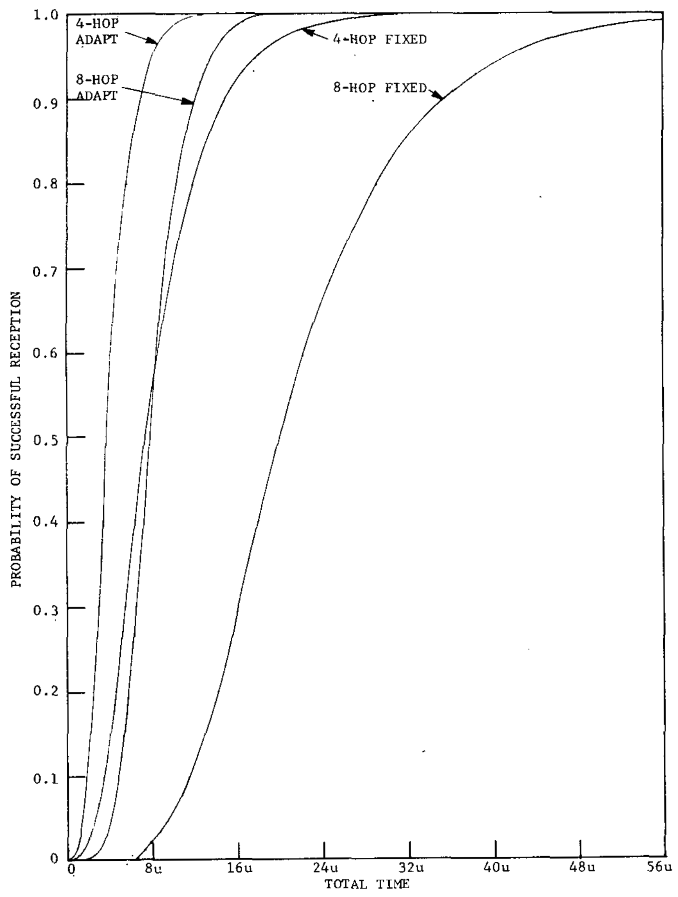


Fig. 7. Comparison of message delivery time for a fixed rate and adaptive code rate system.

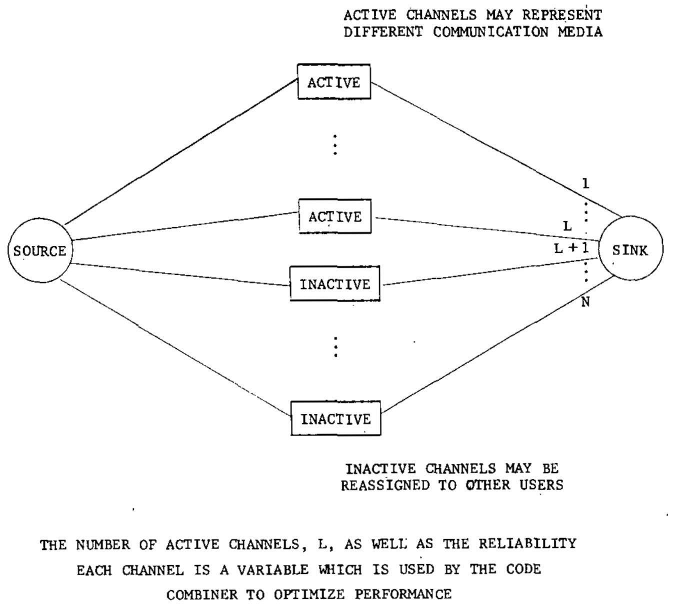

ACTIVE CHANNELS MAY REPRESENT DIFFERENT COMMUNICATION MEDIA
ACTIVE
ACTIVE
INACTIVE
INACTIVE
SOURCE
SINK
1
L
L + 1
N
INACTIVE CHANNELS MAY BE REASSIGNED TO OTHER USERS
THE NUMBER OF ACTIVE CHANNELS, L, AS WELL AS THE RELIABILITY EACH CHANNEL IS A VARIABLE WHICH IS USED BY THE CODE COMBINER TO OPTIMIZE PERFORMANCE

Fig. 8. Parallel channel applications.

time per link can only enhance the adaptive code combining performance.

F. One-Way Broadcast Applications

One approach of broadcasting a message to a group of terminals within a network is to allow each terminal to continually repeat the same message as soon as it is received. A time-out may be contained within the message to allow rebroadcasting of an existing message and to ensure that the network is clear when a new message is to be broadcasted. An approach of this nature is sometimes called flooding, since as time increases more terminals are "flooding" the same message throughout the network. It is assumed that if a terminal receives transmissions of the same message from more than one source, the signals enhance each other rather than interfere with each other. An energy combining receiver satisfies this assumption.

G. Parallel Channel Applications

This final application illustrates that a variable decoding time is not necessarily required to implement code combining.

Consider a link with several active channels that contain the same information. These could be several frequency assignments within a communication band, or even transmissions over completely different channels, e.g., satellite, HF, and a microwave relay. The ultimate spread-spectrum system not only spreads over a given frequency band, but also spreads over multiple communication media.

The code combiner weights the data by the reliability of each communication channel, to arrive at reliable data. The inactive channels shown in Fig. 8 are used to ensure enough redundancy for reliable communications. For network applications these inactive channels may be reassigned to other source-sink pairs.

It is interesting to note that when code combining is transmitted in parallel over multiple communication channels, the data can be optimally combined by the code combiner, rather than have the operator select the "best" communication channel.

IV. Conclusions

Code combining represents a technique of combining noisy packets to achieve error-free results for all channels with finite capacity, i.e., for channels with raw bit errors below 50 percent. Also, code combining minimizes the decoding delay by allowing a receiver to combine the minimum number of packets needed to obtain correct data. Features such as these should prove to be useful for a wide range of applications such as indicated in this paper.

References

[1] A. J. Heller and I. M. Jacobs, "Viterbi decoding for satellite and space communication," *IEEE Trans. Commun. Technol.*, vol. COM-19, p. 835, Oct. 1971.

[2] J. A. Heller, "Communications systems research," JPL Space Progr. Summary, vol. III, pp. 37-54.

[3] W. W. Peterson and E. J. Weldon, Jr., *Error Correcting Codes*, 2nd ed. Cambridge, MA: M.I.T. Press, 1972.

[4] R. A. Scholtz, "The spread spectrum concept," *IEEE Trans. Commun.*, vol. COM-25, Aug. 1977.

[5] D. Chase and L. H. Ozarow, "Capacity limits for binary codes in the presence of interference," *IEEE Trans. Commun.*, vol COM-27, p. 441, Feb. 1979.

[6] E. J. Weldon, "An improved selection repeat ARQ strategy," *IEEE Trans. Commun.*, vol. COM-30, pp. 480–486, Mar. 1982.

[7] P. S. Sindhu, "Retransmission error control with memory," *IEEE Trans. Commun.*, vol. COM-25, pp. 473–479, May 1977.

[8] Y.-M. Wang and S. Lin, "Modified selective repeat type II hybrid ARQ system: Performance analysis," *IEEE Trans. Commun.*, vol. COM-31, pp. 593–608, May 1983.

[9] J. J. Metzner and D. Chang, "Efficient selective repeat ARQ strategies for very noisy channels," in *Conf. Rec. IEEE GLOBECOM*, Dec. 1983, pp. 35.2.1–35.2.8.

★

David Chase (S'67–M'68–SM'76) received the B.E.E. degree (magna cum laude) from the City College of New York, New York, NY, in 1962 and the S.M. and Ph.D. degrees from the Massachusetts Institute of Technology, Cambridge, in 1964 and 1967, respectively.

From 1967 to 1972 he was employed by General Atronics Corporation, a subsidiary of the Magnavox Company, Philadelphia, PA, as a Staff Engineer and finally as Manager of the Communications Research Department. During this period he also served as a Lecturer for the Graduate School of the Pennsylvania State University, King of Prussia. At present, he is President of CNR, Inc., and has actively been involved in its technical direction since its inception in 1972. His areas of current interest include error control coding, modem and waveform design, antijam communications systems, networking, and channel characterization.

Dr. Chase is a member of Eta Kappa Nu, Tau Beta Pi, and Sigma Xi.