CODE COMBINING

David Chase

CNR, Inc.
220 Reservoir Street
Needham, MA 02194

ABSTRACT

It is well-known, that by choosing the data rate below the channel capacity, error-free communications is possible. Furthermore, existing practical error-correction coding techniques can be used to meet reliability constraints. However, a basic design problem for reliable digital communications is the selection of the code rate. The popular rate-1/2 code is a good choice for AWGN channels, but is a poor choice for channels where a large fraction of the transmitted bits are destroyed by interference. Designing a system for the worst-case scenario may require extremely low rate codes to bring the data rate below channel capacity (ref. 1). Code combining is an adaptive technique, which matches the code rate to the prevailing channel conditions.

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 is, at best, an educated guess. Some examples are channels with non-stationary noise, such as the wideband 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.

CODE COMBINING ELEMENTS

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%. Two critical features of code combining are:

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

(ii) Weighting of each packet by an estimate of its reliability (soft decision on packets).

Figure 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 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 may be added to the information packet so that the decoder can determine when to stop combining packets. 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 Figure 1, a code rate of R/L is assumed to be sufficient for successful decoding.

The code combiner treats the received packets $\underline{Y}^1$, $\underline{Y}^2$, ..., $\underline{Y}^L$ as a code of rate R/L and attempts to pick the most probable information packet. 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 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. 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.

The minimum free distance of a convolutional code is upper bounded (ref. 2) by

$$d' \le \min_h \frac{2^h}{2^h - 1} \frac{K+h-1}{2R} \qquad (1)$$

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 1 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, the key to code combining is the use of a maximum-likelihood decoder. As a simple example the minimum distance of the repeated Golay code is listed on Table 2. 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% for this repeated Golay code. As is shown in the Performance Examples section, 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

$$X_i = \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1 - p \end{cases} \qquad (2)$$

The number of errors in a codeword of length N is

$$\alpha = \sum_{i=1}^{N} X_i \qquad (3)$$

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 of error is

$$p(\varepsilon) = \Pr[\alpha > e] = \Pr\left[\frac{1}{N}\sum_{i=1}^{N} X_i > \frac{e}{N}\right] \qquad (4)$$

4.2.2

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

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

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

$$p(e) \rightarrow 0 \quad \text{for } p < \frac{1}{6}\ (16.7\%)$$

$$p(e) \rightarrow 1 \quad \text{for } p > \frac{1}{6}\ (16.7\%) \qquad (6)$$

Similarly for the best block codes we have upper and lower bounds on the minimum distance (ref. 3), which state that as $R \rightarrow 0$

$$\frac{d}{N} \rightarrow \frac{1}{2} \qquad (7)$$

Thus, for any bounded distance decoder we must have

$$\frac{e}{N} \leq \frac{1}{4} \qquad (8)$$

so that

$$p(e) \rightarrow 1 \quad \text{for } p > \frac{1}{4}\ (25\%) \qquad (9)$$

The above arguments show that bounded distance algebraic decoders cannot operate for $p > 25\%$, however, the positive channel capacity ($C = 0.1887$, at $p = 25\%$) indicates that error-free communications 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 $p < 0.50$.

Packet Format

A straightforward technique for implementing code combining is to use a packet format composed of three components illustrated in eq. 10

$$\underline{P} = [\text{sync}|\text{data}|\text{CRC}] \qquad (10)$$

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.

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 are used by the 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. On Figure 1 the information packet $\underline{I}$ represents the data and the CRC bits. The synchronization bits represent a one-bit code (assuming sync bits are inverted for each packet change) which can also be code combined over successive packets.

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"* 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 unavailable when a fixed low-rate coding approach is used.

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 $i = 1, 2, \ldots, L$. A maximum-likelihood decoder will select the codeword (packet) m which maximizes the conditional probability between the received sequence $\underline{Y}$ and the repeated packet denoted as $\underline{X}_m$. This function can be written as

$$\max_m \left\{ p[\underline{Y}|\underline{X}(m)] = \prod_{i=1}^{L} (1-p_i)^{N-d_i(m)} \, p_i^{d_i(m)} \right\} \tag{11}$$

where $d_i(m)$ is the number of bit disagreements for the ith packet. Taking the natural logs of both sides yields

$$\max_m \left\{ \ln p[\underline{Y}|\underline{X}(m)] \right\} = \max_m \left[\sum_{i=1}^{L} N \ln(1-p_i) - d_i(m) \ln\left(\frac{1-p_i}{p_i}\right) \right] \tag{12}$$

Finally, by dropping the first term on the left which is not a function of m and omitting the negative sign by taking a min rather than max, we obtain

$$\max_m \left\{ \ln p[\underline{Y}|\underline{X}(M)] \right\} \propto \min_m \sum_{i=1}^{L} d_i(m) \ln\left(\frac{1-p_i}{p_i}\right) \tag{13}$$

*Channel error rates approaching 50%.

**This 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.

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

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

Some typical weights are listed in Table 3. 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. Another approach is to record a number of errors a decoder would correct when operating on an uncombined packet. Information from the demodulator may also be used to obtain an estimate of p_i.

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

$$C = 1 - p \log_2 p - (1-p)\log_2(1-p) \tag{15}$$

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

$$N_p = \frac{1}{2C} \tag{16}$$

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

$$R \le C = 0.2781 \tag{17}$$

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

$$N_p \ge \frac{1}{2C} = 1.7981 \tag{18}$$

4.2.4

For a decoded error rate of 10^{-4}, and $p = 0.2$, Figure 2 illustrates that about 5 packets are required for a $K = 7$ code, about 6 packets for a $K = 5$ code, and about 7 packets for the (24,12) code. These curves were obtained by actual simulations but union bounds may be used to provide adequate estimates in the 10^{-4} region.

The important point to note is that for all $p < 0.5$, error-free communications 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. Figure 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.

APPLICATIONS

A 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. The concepts, rather than the details needed to actually implement a system, are highlighted.

Spread Spectrum Systems

For a fairly wide class of spread spectrum systems (ref. 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

$$\frac{E_b}{(N_0)_{eq}} = \frac{S}{DN_0 + J\left(\frac{D}{W_s}\right)} \tag{19}$$

The factor W_s/D, called the processing gain, represents the attentuation 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 combatting this uncooperative jammer is to use a lower rate error-correcting 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.

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.

Two-Way Links (ARQ Code Combining)

Perhaps the simplest application of code combining is in an automatic repeat request (ARQ) two-way communication system. Data is 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.

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

Assuming the transmission delay is less than a packet an odd/even packet format can be used to allow a continuous flow of data based on the feedback information as shown on Figure 4. Generalization for links with delays of multiple packets is straightforward.

Each information 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 receives 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.

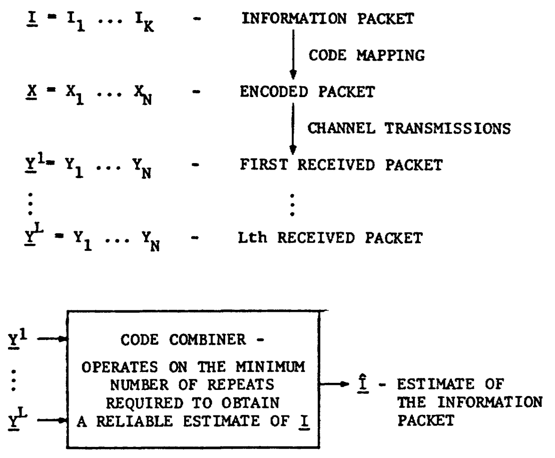
In the ARQ system, the reliability of the return ACK/NACK must be considered, therefore, the ACK/NACK will be much more heavily coded than the basic data packet rate. An unreliable ACK/NACK can be interpreted as a NACK which forces the transmitter to continue to repeat the current packet. This is the 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.

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%. 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. David Chase and Lawrence H. Ozarow, "Capacity Limits for Binary Codes in the Presence of Interference," IEEE Trans. on Communications, Vol. COM-27, No. 2, February 1979, p. 441.
2. J.A. Heller, Communications Systems Research," JPL Space Programs Summary 37-54, Vol. III.
3. W.W. Peterson and E.J. Weldon, Jr., Error Correcting Codes, 2nd Edition, MIT Press, 1972.
4. R.A. Scholtz, "The Spread Spectrum Concept," IEEE Trans. on Comm., Vol. COM-25, No. 8, August 1977.



I = I1 ... IK - INFORMATION PACKET
CODE MAPPING
X = X1 ... XN - ENCODED PACKET
CHANNEL TRANSMISSIONS
Y1 = Y1 ... YN - FIRST RECEIVED PACKET
YL = Y1 ... YN - Lth RECEIVED PACKET
Y1
YL
CODE COMBINER - OPERATES ON THE MINIMUM NUMBER OF REPEATS REQUIRED TO OBTAIN A RELIABLE ESTIMATE OF I
Î - ESTIMATE OF THE INFORMATION PACKET

Figure 1. Sequence of Mappings Used in Code Combining

TABLE 1. 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

TABLE 2. 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

TABLE 3. Values for the Optimum Weighting Factor

P_i	$\ln\left(\frac{1 - P_i}{P_i}\right)$
0.5	0.000
0.4	0.405
0.3	0.847
0.2	1.386
0.1	2.197
0.05	2.944
0.01	4.595
0.005	5.293
0.001	6.907
0.0005	7.600
0.0001	9.210

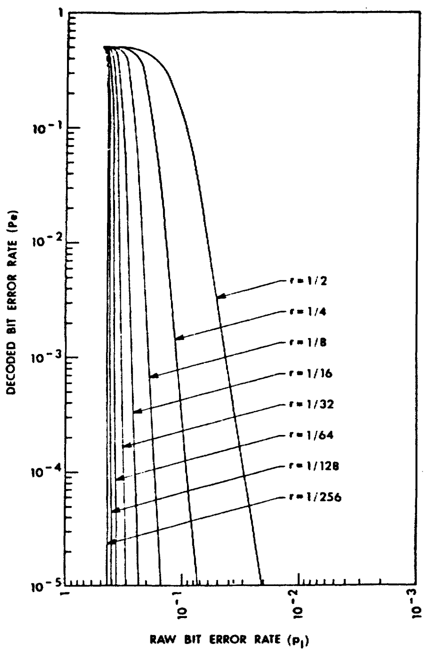


Figure 3. Simulated Performance of K = 7 Convolutional Codes with Hard Decoding and Independent Bit Errors

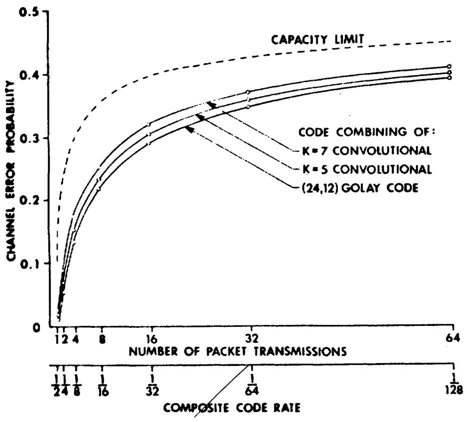


Figure 2. Comparison of Code Combining and Capacity Limits

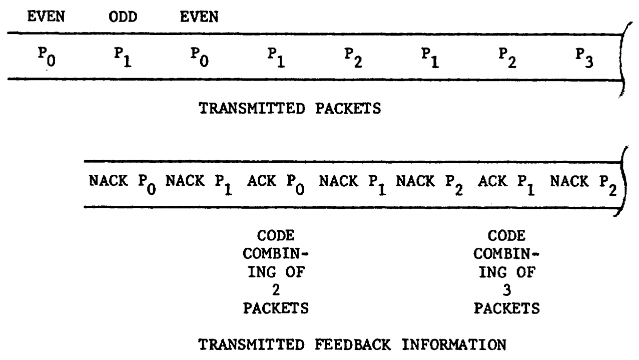

EVEN
ODD
EVEN
P0 P1 P0 P1 P2 P1 P2 P3
TRANSMITTED PACKETS
NACK P0 NACK P1 ACK P0 NACK P1 NACK P2 ACK P1 NACK P2
CODE COMBINING OF 2 PACKETS
CODE COMBINING OF 3 PACKETS
TRANSMITTED FEEDBACK INFORMATION

Figure 4. Odd/Even Packet Scheme with ACK/NACK

4.2.7