Error Detection and Correction

This lecture introduces the topic of codes that allow for error detection and correction, also called channel codes.
After this lecture you should be able to: list some advantages and disadvantages of checksums, compute even and odd parity bits, compute the Hamming distance between two code words, compute the code rate for block, punctured and non-punctured convolutional codes, correct errors in a received block code word by exhaustive search, compute coding gain, and compute the punctured output of a convolutional encoder.
Updated Nov. 13 to correct the error detecting and correcting ability of a block code.

Checksums

A simple way to check for errors in a frame of data is to compute the sum of the byte (or word) values in a frame of data. The sum is computed modulo the word size of the check sum. The additive complement of the sum is then appended to the packet so that an error-free packet has a zero checksum. This is the type of error-detection used by TCP/IP frames used on the Internet.

Checksums are typically used by higher-level protocols since they are easy to compute in software. However, there are many common types of errors, such as insertion of zeros and transposition of values that are not detected by checksums.

Exercise 1: Compute the modulo-4 (3-bit) checksum of a frame with values 4, 1, and 3. Would an error be detected if the received frame was 4, 1, 3, 0? How about if the received frame was 1, 4, 3?

Error Detection

Another technique for detecting errors in received frames is for the transmitter to compute additional bits called "parity bits" and add them to the end of the frame. The receiver computes the parity bits in the same way using the received data and compares them to the received parity bits. If the transmitted and received parity bits match then either there were no errors or the errors were such that the errors resulted in the same parity.

The probability of the latter event is called the undetected error probability. Good error detecting codes try to make this probability as low as possible.

Parity Bits

The simplest type of parity is a single parity bit. Typically the parity bit is computed as the modulo-2 sum of all of the bits in the message.

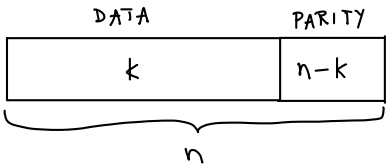
Exercise 2: What is a modulo-2 sum? What is the modulo-2 sum of 1, 0 and 1? What is the modulo-2 sum if the number of 1's is an even number?

The most common example of a single parity bit is a parity bit added to each ASCII character. Most serial interfaces can be configured to compute and append a parity bit to each ASCII character. This bit can be either the sum of all of the bits ("even parity") or it's complement ("odd parity").

The receiver computes the parity bit from the data bits and compares the computed parity bit to the transmitted parity bit. If the computed and received parity bits match then there was either no error or there were an even number of errors.

Block Codes

More complex channel codes use multiple parity bits. A block code where each block of n bits contains k data bits and $n-k$ parity bits is called an (n,k) code:



Exercise 3: How many different code words (different blocks) does an (n,k) code have? How many different patterns of $n-k$ parity bits are there?

The *Hamming Distance* is the number of bits that differ between two code words. The performance

of a code is generally determined by the minimum (Hamming) distance between code words.

Exercise 4: What is the Hamming distance between the codewords 11100 and 11011?

The *rate* of a code is the ratio of information bits to total bits, or k/n. This is a measure of the efficiency of the code. Typically there is a tradeoff between the error-correcting ability (minimum distance) and the rate of a code.

Forward Error Correction Coding

Certain block codes allow the receiver to correct errors introduced by the channel. These types of codes are called Forward Error Correcting (FEC). Error correction is possible when the code words include enough parity bits that the receiver can "guess" which codeword was transmitted even though the received codeword does not match any of the codewords that could have been transmitted (in other words, it contains errors).

Minimum Distance Decoding

Conceptually, a receiver can correct errors by choosing the valid codeword that has the smallest Hamming distance from the received codeword. This is because codewords with fewer errors are more likely to happen than those with more errors.

Exercise 5: A block code has two valid codewords, 101 and 010. The receiver receives the codeword 110. What is the Hamming distance between the received codeword and each of the valid codewords? What codeword should the received decide was sent?

In general, a block code with a minimum Hamming distance between valid codewords of d can detect $d-1$ errors and correct $\lfloor \frac{d-1}{2} \rfloor$ where $\lfloor \cdot \rfloor$ denotes the floor function (round down to the next smallest integer).

Howver, with large codewords it is not possible to do a simple exhaustive search through all possible codewords to find the one with the minimum distance. There is a large field of study devoted to the design of codes which can be efficiently encoded and decoded.

Coding Gain

When the FEC code is well-matched to the error rate and to the types of errors likely to be encountered, the use of FEC results in higher throughput and better power efficiency.

Higher throughput results because blocks that contain correctable errors do not need to be retransmitted.

Exercise 6: Assume errors on a channel cause a frame error rate of 50%. When a rate-1/2 FEC code is used the frame error rate drops to 2%. Compute the throughputs with and without coding relative to the uncoded and error-free channel. What other advantages might the use of FEC provide?

Better power efficiency results if the same post-correction error rate can be achieved by transmitting less energy per information bit (E_b). This reduction in the required E_b to achieve a certain error rate is called the "coding gain".

Exercise 7: A system without coding needs to transmit at 1W to transmit 1 Mb/s at an error rate of 10^{-3}. When a rate-1/2 code is used the power to transmit the necessary 2Mb/s of data and parity bits decreases to 500mW. What is the channel bit rate in each case? What is the information rate in each case? What is E_b? What is the coding gain?

Convolutional Codes

Although it is possible to use block codes to implement FEC, the trend until recently has been to use convolutional codes for communication systems. This was mainly because of the existence of a relatively simple and efficient decoding algorithm for convolutional codes called the Viterbi algorithm.

A rate k/n convolutional code is implemented by reading a certain number of bits into a shift register and outputting a number of modulo-2 sums of these bits as shown below[1]

[1] Taken from the 802.11 standard.

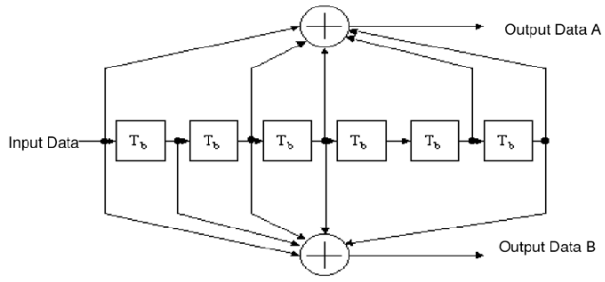


Figure 18-8—Convolutional encoder (k = 7)

If there are n output bits for each k input bits the rate of the code is k/n. The "constraint length," (K) of the code is the number of bits that can affect each output bit and is equal to the length of the shift register plus one (because the input bit is also used in computing the output).

Exercise 8: Assuming one bit at a time is input into the encoder in the diagram above, what are k, n, K and the code rate?

Viterbi Algorithm

Most FEC decoders use an algorithm called the Viterbi algorithm. This algorithm is a "maximum likelihood" decoder because it chooses the bit sequence with the minimum distance from the received sequence (or close to it).

The VA uses the trellis structure of the code (a tree with branches that merge) to avoid exponential increase in decoding complexity with message length. Instead, the complexity is proportional to 2^K where K is the constraint length. Although the algorithm is relatively complex, implementations are readily available.

Although many different convolutional codes are possible, there are certain standard codes that are used by many different systems. Hardware implementations are typically only available for these codes. The most common convolutional code is the rate-1/2 code with a constraint length of 7 shown above.

Higher-rate codes can be derived from the basic rate-1/2 code by not transmitting some of the bits. This is called "puncturing." The same decoder hardware can be used by feeding the decoder a value representing "unknown" (an "erasure") in place of the missing parity bits.

Exercise 9: Consider the encoder above. If the only the bits corresponding to the outputs A, A and B, and B are transmitted corresponding to every three input bits, what is the code rate of this punctured code?

Modern FEC Codes

Two other FEC codes have become popular in recent years because their error-correcting performance approaches the Shannon Limit.

Turbo Codes use two different codes to encode the data. The decoder uses the information from one code to help with decoding of the other code. Then that information is used to help decode the first code. The procedure is repeated iteratively until there are no errors (determined by a CRC) or a limit is reached.

Low Density Parity Check (LDPC) codes are block codes with sparse (few 1's) parity check matrices. This means that each parity bit is a function of only a few message bits.

Most FEC decoders use "soft-decision" decoding algorithms that operate on the probabilities that particular bits are zeros or ones rather than operating on binary "hard decisions".

Reed Solomon Codes

The Reed-Solomon code is a block FEC code that is widely used. RS codes operate on non-binary Galois fields, typically GF(256). It is able to correct a certain number of 8-bit word errors, regardless of the number of bit errors in each word. For this reason Reed-Solomon codes are efficient for channels that have bursty errors that cause multiple errors to fall within the same 8-bit word.