Lecture 4: Wireless Physical Layer: Channel Coding

Mythili Vutukuru

CS 653 Spring 2014

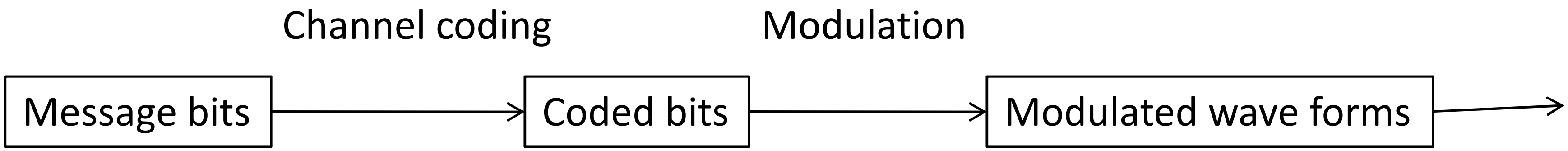
Jan 16, Thursday

Channel Coding

- Modulated waveforms disrupted by signal propagation through wireless channel – leads to errors when recovering bits from signal
- Channel coding – add redundancy in the transmitted bits, so that you can detect and correct errors
- Error detection – received coded bits indicate that bit errors have occurred
- Error correction – received coded bits indicate that bit errors have occurred, and help to identify and correct the location of errors

Modulation and Channel Coding

- The modulation and coding schemes decide the bit rate of transmissions over the air.
- Wise choice of modulation and coding schemes helps reach Shannon capacity
- Message bits are converted into coded bits before modulation at transmitter
- At receiver, the received waveform is demodulated into bits, then the channel code is decoded to recover the original message bits



Simple example – repetition coding

- A very simple channel coding scheme is repetition coding – repeat each bit multiple times, so that you can recover the message even if some bits are in error
- For bit “0” – send “000”
- For bit “1” – send “111”
- “000” and “111” are called valid codewords
- Received codeword should ideally be 000 or 111. In reality, due to errors, it can be any 3 bit string, say 001
  - If we assume only 1-bit errors, we can detect and correct the error and say that the transmitted code word was 000
  - If we assume 1 or 2 bit errors, then we know there was an error, but we can’t be sure what the transmitted codeword was (only error detection, no correction)
- That is, this simple code can detect up to 2 bit errors, or correct single bit errors

Hamming distance

- Hamming distance of two bit strings is defined as the number of bits that are different in the two strings
- Example, hamming distance between 000 and 101 is 2
- For a given code, take all possible codewords, find the hamming distance between every pair. The minimum distance between any two codes is also called the minimum hamming distance of the code.
- If a code has min. hamming distance *d*
  - It can detect *d-1* errors, or
  - correct *floor{(d-1)/2}* errors
- Example, the repetition code 000,111 has a minimum hamming distance of 3, so detect 2 errors or correct 1 error.

Common channel codes

- Usually physical layer designs use an error correcting code on the message bits, try to recover as many errors as possible.
- Types of error correcting codes
  - Block codes: a block of message bits is mapped into a code word
  - Convolutional codes: a stream of message bits is mapped into a stream of coded bits continuously.
  - More complicated: turbo codes, LDPC codes etc. We will not cover these
- Most wireless physical layers use convolutional codes, moving to turbo codes and other advanced codes recently
- This lecture will only provide a brief introduction to simple block and convolutional codes

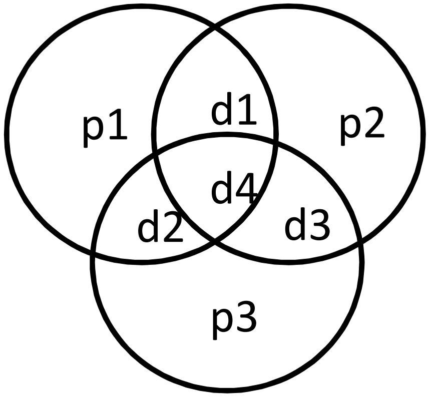
Common channel codes (2)

- Still, some errors may be undetected after error correction (for example, you assume single bit error and correct 001 to 000, whereas it was actually a two bit error from 111)
- Physical layers also use an error detecting code over the entire packet to catch these rare cases
- Simplest error detecting code – add a parity bit
  - Parity bit is modulo 2 sum or exor of all bits. That is, if number of “1”s is even, parity is 0, else it is 1
  - Example, parity of “001110” is 1, parity of “001100” is 0
- Receiver recomputes parity. If parity doesn’t match, 1 error (or 3 or 5 or…) has occurred. Parity cannot detect even number of errors.
- Better than parity – check sum (a better type of parity sum over bits). Used to detect errors in packet headers etc.
- Most common error detection in wireless physical layers – CRC (cyclic redundancy check). CRC is computed over entire packet and added to the end of a packet while transmission. Receiver checks CRC. If check fails, it means packet has some uncorrected bit errors (cannot point to where the error is, and cannot correct it, only detect it)

Block codes

- The repetition code is a simple example of a block code.
- In a block code, if *k* message bits are mapped into an *n*-bit code word, such that the minimum hamming distance of the code is *d*, then the block code is denoted by *(n,k,d)*
- The rate of the code *R* = *k/n*, denotes how efficient the code is in using the channel capacity
- Ideally, we want codes with large *R* (so that very little extra bits are sent) but also large *d* (so that many errors are corrected)
- Repetition code (repeated c times) is a (c,1,c) block code
- Block codes are not very common in wireless physical layers. We will only cover a simple example next and move on to convolutional codes.

Hamming code



- The following is an example of a good block code.
- Hamming code (7,4,3)
- Consider data bits d1, d2, d3, d4
- Then we add three parity bits to the 4 bits to get a 7 bit code word.
- The parity bits are computed as follows, as depicted by the figure (all additions are modulo 2, or exor operations)
  - p1 = d1 + d2 + d4
  - p2 = d1 + d3 + d4
  - p3 = d2 + d3 + d4
- When the code word is received, the receiver recomputes the parity bits, and corrects single bit errors in the 4 data bits as follows
  - If p1 and p2 don’t match, d1 is in error, flip to correct
  - If p1 and p3 don’t match, d2 is in error, flip to correct
  - If p2 and p3 don’t match, d3 is in error, flip to correct
  - If p1, p2, p3 don’t match, d4 is in error, flip to correct
  - If only one of p1 or p2 or p3 is in error, the parity bit itself is in error, so data bits are fine

Convolutional codes

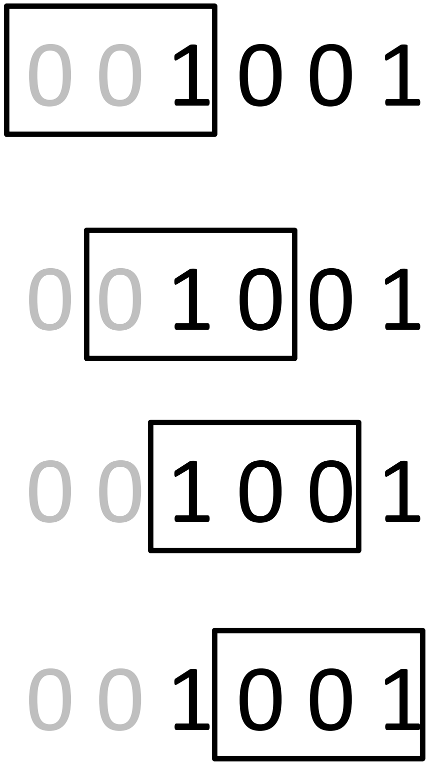
- Convolutional codes operate on a stream of bits.
- In each step of the computation, *k* bits of the message are consumed, and *n* coded bits are produced
- The coded bits are computed using not just the current set of *k* bits, but the last *l* such sets of *k* bits
- The number *l* is called the constraint length of the code
- The rate of the code is $R = k/n$
- Intuitively, convolutional codes work by distributing the load of error correction over the entire stream, instead of over localized blocks (as in block codes). Therefore, codes with larger constraint length lead to better error correction
- For example, most WiFi physical layers use a convolutional code with rate ½ and constraint length 7.

Convolutional codes - example

- Consider the following code, *k* =1, *n* = 2, *l* = 3
- That is, in each step, one bit is read, a window of last 3 bits is considered (including the current bit), and 2 coded bits are produced.
- Let the stream of bits be denoted by *x[i]* (index *i* goes over all bits in the message)
- When we read in each bit *x[i],* we compute two parity bits *p[i]* and *q[i]* as follows (all additions are mod 2)
  - *p[i] = x[i] + x[i-1] + x[i-2]*
  - *q[i] = x[i] + x[i-1]*
- The parity bits can also be denoted by a simpler representation called generators. The generator for *p* = 111, *q* =110
  - Here the generators indicate which bits in the last 3 bit window are added to produce this parity bit
- For the first bit (i=0), assume *x[i-1] = x[i-2] = 0*
  - Usually the first *l-1* bits of a convolutional code are hardcoded to a known value to start the coding process from the first bit
- Let us walk through a simple encoding process with this code

Convolutional codes – Encoding by example

- Message = *1001*
- *i=0*. We read the first bit "1" of the message (assume the last 2 bits are 0 to bootstrap)
  - $p[0] = 1 + 0 + 0 = 1$
  - $q[0] = 1 + 0 = 1$
- *i=1* We read the next bit "0"
  - $p[1] = 0 + 1 + 0 = 1$
  - $q[1] = 0 + 1 = 1$
- *i=2* We read the next bit "0"
  - $p[2] = 0 + 0 + 1 = 1$
  - $q[2] = 0 + 0 = 0$
- *i=3* We read the next bit "1"
  - $p[3] = 1 + 0 + 0 = 1$
  - $q[3] = 1 + 0 = 1$
- Finally, we send all parity bits 11111011 and discard the original bits



0 0 1 0 0 1

0 0 1 0 0 1

0 0 1 0 0 1

0 0 1 0 0 1

Convolutional codes - decoding

- The receiver gets the encoded bits (presumably with bit errors). Must recover the original message. How? Not so simple as block codes
- In the previous example, receiver gets 8 bit string. A naïve way to decode:
  - Consider every possible 4 bit message string (16 of them)
  - Encode it using the code to get a 8 bit coded string
  - Compare these 16 coded strings to the received string. Find the one with the minimum hamming distance
  - Identify the message string corresponding to this minimum hamming distance as the likely transmitted message
- Complexity of this algorithm is exponential in length of message
- In real life, a much more efficient algorithm called Viterbi algorithm is used. It follows the same principle, but implements it in a much smarter way, such that the complexity is only linear in terms of the message length
- We don't have time to cover Viterbi algorithm in the course, you can look it up online if you like

Another technique: interleaving

- Channel codes work well with random errors, that is, bit errors are distributed randomly across a packet (so that each codeword has a small number of errors that can be corrected)
- However, wireless channels produce burst errors, that is, errors are concentrated in one part of the packet where channel fading was high etc.
- Interleaving is a technique used to convert burst errors to somewhat random errors
- Interleaving shuffles the bits in a packet after coding. That is, bits of one codeword are not placed contiguously, but at different parts of a packet. This way, a burst error is unlikely to effect all bits in a codeword, and error correction can still work.
- A simple example of interleaving of 12 block-coded bits (4 codewords of 3 bits each) is shown below. All bits of the same color belong to the same codeword.