College of Computer Science
Northeastern University
Wireless Networks CS G250

Fall 2008
October 31 2008

Problem Set 2 - Solutions

1. **Codes**

   (a) Let's look at the ball representation for Hamming codes (Figure 2). The total number of strings is 2^n. In each ball we have $n+1$ strings: one codeword (what you want to send) and n words (all the strings that have exactly one bit flip from the codeword). Given the fact that the Hamming code is a perfect linear code, that means the balls are not overlapping and that there is no point left outside any ball. Then the total number of balls is $\frac{2^n}{n+1}$. The total number of balls is an integer, meaning that $n+1$ must divide 2^n. The only way in which this is true, is when $n+1$ is a power of 2. So, $n+1 = 2^t$, where t is some integer. Then $n = 2^t - 1$.

   k is the number of data bits you want to send. Remember that in each ball we have one and exactly one codeword. Then the total number of balls is equal to the total number of codewords of k bits. So, $2^k = \frac{2^n}{n+1}$. However, $n = 2^t - 1$. That means $2^k = \frac{2^{2^t-1}}{2^t-1+1} = \frac{2^{2^t-1}}{2^t} = 2^{2^t-t-1} \Rightarrow k = 2^t - t - 1$.

   (b) The $2^t - 1 \times t$ matrix is the following:

$$H = \begin{pmatrix} 0 & 0 & 0 & ... & 0 & 0 & 1 \\ 0 & 0 & 0 & ... & 0 & 1 & 0 \\ 0 & 0 & 0 & ... & 0 & 1 & 1 \\ & & & ... & & & \\ 1 & 1 & 1 & ... & 1 & 1 & 1 \end{pmatrix}$$

   H has $2^t - 1$ rows and t columns.

   We need to show two things:

   - any string of length $2^t - 1$ is either a codeword or at distance 1 from a codeword

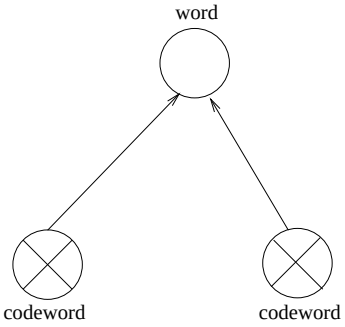

Figure 1: This situation is not possible for the Hamming codes

- there is no overlapping between the balls. In other words the situation described in Figure 1 is not possible.

For every codeword the product between the codeword and H is a vector of all 0s. In other words,

$$c \times \begin{pmatrix} 0 & 0 & 0 & ... & 0 & 0 & 1 \\ 0 & 0 & 0 & ... & 0 & 1 & 0 \\ 0 & 0 & 0 & ... & 0 & 1 & 1 \\ & & & ... & & & \\ 1 & 1 & 1 & ... & 1 & 1 & 1 \end{pmatrix} = Z$$

Each codeword c is a vector of size $n = 2^t - 1$ and Z is a vector of size t. Remember that by codeword we mean the string we want to send, with no errors.

For any string X we have two possibilities. It is either a codeword, that means $c \times H = Z$ and we are done. Either $c \times H = W$, where W is not all 0s. Let's suppose the decimal representation of vector W is w. Then w corresponds to the w^{th} row of matrix H. If we flip bit w in string X, then the product between X with bit w flipped and H will be Z. Thus, any string X is either a codeword or it is at distance 1 from a codeword.

Now let's look at the situation described in Figure 1. Let's suppose we have two different codewords: C_1 and C_2. We also have a word W. In other words, from C_1 to W there is a bit flip i. From W to C_2 there is another bit flip j where i and j are not the same bit. If i and j are the same bit, that implies C_1 and C_2 are identical, which contradicts our assumption. $C_1 \times H = Z$. By the above construction, C_2 is obtained from C_1 by flipping

two different bits i and j. So, if we flip 2 bits i and j in C_1, then the result will be a vector that is the sum of row i and row j of matrix H. Given the fact that i and j are different, then the sum of row i and row j won't cancel out. It other words, the result of $C_2 \times H$ is not a vector of all 0s. That means C_2 is not a codeword. Which is a contradiction. Therefore the situation depicted in Figure 1 can not happen.

(c) The matrix in this case is:

$$G = \begin{pmatrix} 0 & 0 & 0 & ... & 1 & 1 & 1 \\ & & & . & & & \\ & & & . & & & \\ & & & . & & & \\ 0 & 0 & 0 & ... & 1 & 1 & 1 \\ 0 & 0 & 1 & ... & 0 & 1 & 1 \\ 0 & 1 & 0 & ... & 1 & 0 & 1 \end{pmatrix}$$

In this case G acts as a generator, where G has t rows and 2^t columns.

Let's take two strings of size t bits X_1 and X_2. In this case $X_1 \times G = R_1$ and $X_2 \times G = R_2$. We need to show that R_1 and R_2 differ in more than half of the bits. Both R_1 and R_2 are vectors of size 2^t. In other words $X_1 \times G + X_2 \times G = (X_1 + X_2) \times G = Y \times G$, where $X_1 + X_2 = Y$. We need to show that $Y \times G$ has at least $2^t - 1$ ones. For $Y \times G$ to have at least $2^t - 1$ ones, that means there are at least $2^t - 1$ columns that multiplied by Y (in other words $Y \times G1_j$, where G_j is a column of G) give us a 1. Everything is modulo 2. That implies the number of common bits equal to 1 in both G_j and Y is odd. What is the probablity that the intersection between any two random sets G_j and Y has odd cardinality (if we view the strings as sets)? The probablity is half. Therefore X_1 and X_2 differ in more than half the bits.

2. **Hat Problem**

(a) In the case of the 3 hat problem, the players can have the following strategy:

- if a player sees two black hats, the player should say that the color of his hat is white

- if a player sees a black hat and a white hat, then the player will decide not to speak
- if a player see two white hats, the player should say that the color of his hat is black

Given the above strategy, let's see what happens when we enumerate all possible cases:

- Black Black Black
  If you see two black hats, you will say the color of your hat is white. Then you lose.
- Black Black White
  If you see a black and a white hat, then you decide to shut up. Then you win.
- Black White Black
  If you see a black and a white hat, then you decide to shut up. Then you win.
- Black White White
  If you see two white hats, then you say your hat is black. Then you win.
- White Black Black
  If you see two black hats, then you say your hat is white. Then you win.
- White Black White
  If you see a black hat and a white hat, then you decide to shut up. Then you win.
- White White Black
  If you see a black hat and a white hat, then you decide to shut up. Then you win.
- White White White
  If you see two white hats, then you say your hat is black. Then you lose.

In this case, we see that the probablity for 3 students to succeed is at least $\frac{6}{8} = \frac{3}{4}$.

(b) In the case of $2^t - 1$ students, the probability for them to succeed is $\frac{\#wins}{total}$.

The $total$ is 2^{2^t-1}, meaning that this is the total number of ways in which we can assign hats to the $2^t - 1$ students.

The $\#wins = total - \#losses = 2^{2^t-1} - \#losses$. Now let's take a look at $\#losses$.

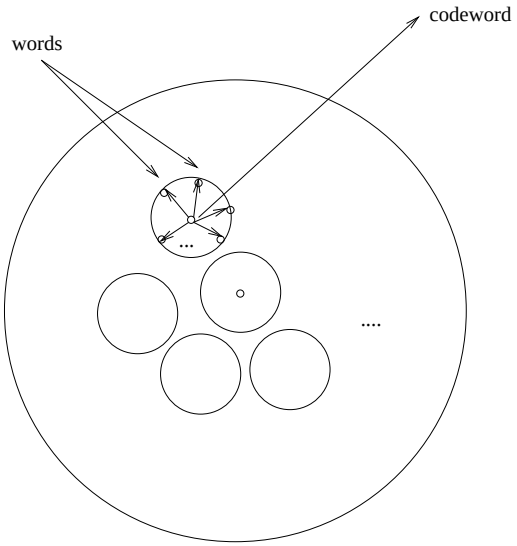

Figure 2: Hamming codes

The hat problem can be reduced to the Hamming codes (Figure 2). A *codeword* is what we are trying to send (or in the context of the hat problem, the correct assignment of hats to the $2^t - 1$ students). *Words* are all the codes we might get if we flip exactly one bit (or in the context of the hat problem, the one bit flip stands for guessing the wrong hat assignment for yourself). The best strategy for the hat problem is that in case you win, everyone guesses correctly. In case you lose, we would like everyone to lose. So we would like all our $2^t - 1$ students to guess incorrectly in case we lose. The number of losses is actually the total number of codewords for which all of the $2^t - 1$ students guess incorrectly the assignment of their hat. Therefore the number of losses is $\frac{2^{2^t-1}}{2^t}$, where 2^{2^t-1} is the total number of codewords and words and 2^t is for one codeword we have $2^t - 1$ possible one incorrect guess (or one bit flip).

$\#losses = \frac{2^{2^t-1}}{2^t}$

Then the probability for 2^t-1 students to succeed is $\frac{2^{2^t-1} - \frac{2^{2^t-1}}{2^t}}{2^{2^t-1}} = 1 - \frac{1}{2^t}$.

(c) Again we will make use of Hamming codes. In this case, for each

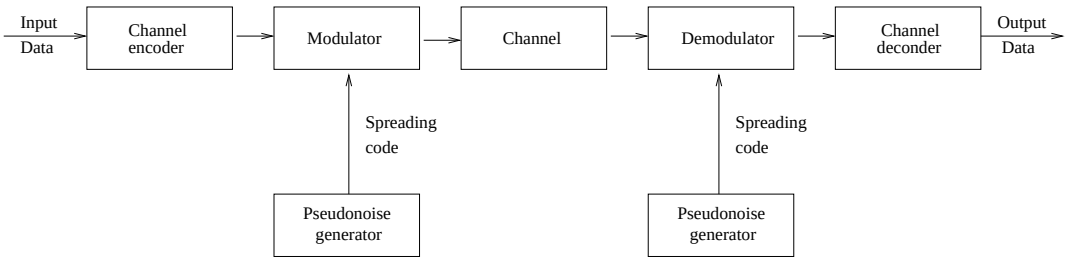

Figure 3: General Model of Spread Spectrum Communication Systems

codeword we have n words.

$\#wins + \#losses = 2^n$, where 2^n is the total number of hat assignments to the n students.

Any strategy can be thought of orienting some edges. A successful node is when you have at least one outgoing and no incoming edges. A losing node can absorbe at most n failures.

Therefore, $\#losses \geq \frac{\#wins}{n} \Rightarrow \#wins \leq n\#losses$

$\#wins + \#losses \leq n\#losses + \#losses = (n+1)\#losses$

But $\#wins + \#losses = 2^n$. That means $2^n \leq (n+1)\#losses \Rightarrow \#losses \geq \frac{2^n}{n+1}$. That implies that the probability of n students to fail is not less than $\frac{\frac{2^n}{n+1}}{2^n} = \frac{1}{n+1}$.

3. **Spread Spectrum**

(a) Spread spectrum techniques are used to transmit either analog or digital data, using an analog signal. The idea is to spread the information signal over a wider bandwidth to make jamming and interception more difficult. Also spread spectrum techniques increase the immunity to noise. The general model for spread spectrum is depicted in Figure 3.

The spreading code or spreading sequence is actually a sequence of digits used to modulate the signal. Typically the spreading code is generated by pseudonoise or pseudorandom number generator. The effect of this modulation is to increase significantly the bandwidth (spread spectrum) of the signal to be transmitted.

Frequency Hopping Spread Spectrum (FHSS) With FHSS the signal is broadcast over a seemingly random series of radio frequencies, hopping from frequency to frequency. The

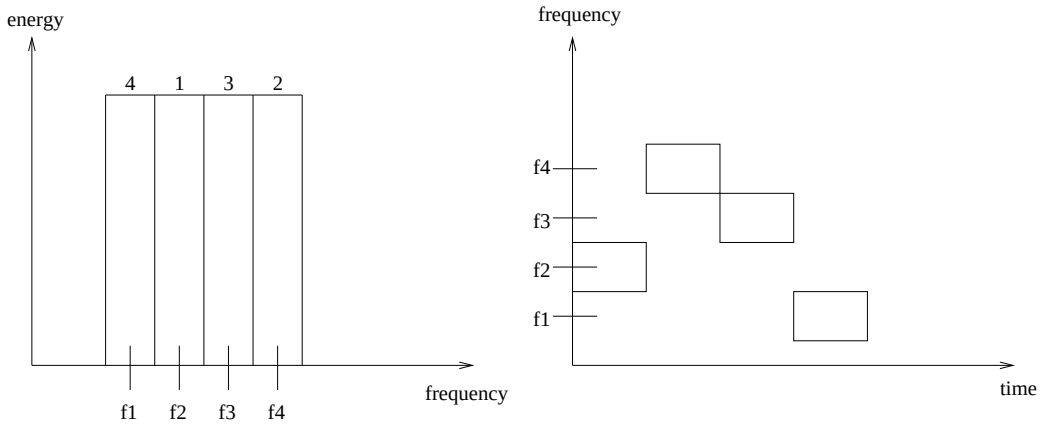

Figure 4: Frequency Hopping Spread Sprectrum - Example

receiver hops between frequencies in synchronization with the transmitter. The eavesdroppers will be able to jam only certain frequencies and therefore catch a few bits. They will be able to intercept the entire message only if they jam the entire spectrum, which requires huge power invested by the eavesdroppers. Both the receiver and the transmitter hop from frequency to frequency in the same order. The transmitter has the responsibility to let the receiver know in which order to hop.

The basic approach for FHSS is to have 2^k carrier frequencies forming 2^k channels. The transmitter operates in one frequency channel for a fixed interval of time and then it hops to a different frequency channel. During the interval, the transmitter sends some number of bits. Figure 4 presents an example of FHSS.

As a fun fact, this technique was invented by Hedy Lamarr, a very famous Hollywood actress in the 40s.

Direct Sequence Spread Spectrum (DSSS) Each bit in the original signal is represented by multiple bits in the transmitted signal. One technique for DSSS is to combine the digital stream with the spreading code bit stream using XOR.

(b) Advantages of spread spectrum:

- less susceptible to interference than other non-spread spectrum systems
- frequency jamming is extremely difficult. It can only be achieved if the jammer knows the pseudo-code and channel allocation. If the jammer doesn't know the pseudo-code and the channel allocation, then all the jammer will get will be a few bits now and then. But still he won't be able to udnerstand the message.
- the spectrum has a very low spectral density
- less susceptibility to multipath fading

Disadvantages of spread spectrum:

- complex circuitry
- very large bandwidths: thus inefficient usage of the bandwidth which is expensive
- expensive to develop

(c) For MFSK, the frequency assignments are centered around f_c, and every symbol is transmitted in a channel of width $2f_d$. The assignments are of the form.

$$f_i = f_c + (2i - 1 - M)f_d$$

giving the following assignments:

000	f_1	=	75 kHz	001	f_2	=	125 kHz
010	f_3	=	175 kHz	011	f_4	=	225 kHz
100	f_5	=	275 kHz	101	f_6	=	325 kHz
110	f_7	=	375 kHz	111	f_8	=	425 kHz

Now we wish to apply slow FHSS to this same scheme with 2 bits of Pseudonoise, which gives 4 possible FH channels. Every FH channel will be centered around a particular f_c^{PN}, starting from the original $f_c = 250$ for the 00 PN sequence, and will be 400 kHz wide:

f_c^{00} = 250 kHz
f_c^{01} = 650 kHz
f_c^{10} = 1050 kHz
f_c^{11} = 1450 kHz

All 32 frequency assignments are:

For PN 00:

000	f_1	=	75 kHz	001	f_2	=	125 kHz
010	f_3	=	175 kHz	011	f_4	=	225 kHz
100	f_5	=	275 kHz	101	f_6	=	325 kHz
110	f_7	=	375 kHz	111	f_8	=	425 kHz

For PN 01:

000	f_1	=	475 kHz	001	f_2	=	525 kHz
010	f_3	=	575 kHz	011	f_4	=	625 kHz
100	f_5	=	675 kHz	101	f_6	=	725 kHz
110	f_7	=	775 kHz	111	f_8	=	825 kHz

For PN 10:

000	f_1	=	875 kHz	001	f_2	=	925 kHz
010	f_3	=	975 kHz	011	f_4	=	1025 kHz
100	f_5	=	1075 kHz	101	f_6	=	1125 kHz
110	f_7	=	1175 kHz	111	f_8	=	1225 kHz

For PN 11:

000	f_1	=	1275 kHz	001	f_2	=	1325 kHz
010	f_3	=	1375 kHz	011	f_4	=	1425 kHz
100	f_5	=	1475 kHz	101	f_6	=	1525 kHz
110	f_7	=	1575 kHz	111	f_8	=	1725 kHz

4. **Shannon's Theorem**

   (a) The noise model for the discrete version of Shannon's Theorem is a noisy channel which flips bits going through it independently with a certain probability: p. The measure of uncertainty in the channel is given by the function $H(p) = -p \log p - (1-p) \log(1-p)$.

   To obtain the expected number of errors in a block of length n, we can define an indicator random variable X_i that will take the value 1 when there is an error on the ith bit, and 0 otherwise. The expected number of errors in the block is then

$$\begin{aligned} E[\text{number of errors}] &= \sum_{i=1}^{n} 1 \cdot p + 0 \cdot (1-p) \\ &= p \sum_{i=1}^{n} 1 \\ &= np \end{aligned}$$

   (b) The number of strings in $\{0,1\}^n$ that have exactly pn 1s can be obtained by observing there are $\binom{n}{np}$ ways to arrange np ones in n positions.

We also need to prove $\binom{n}{np} = 2^{nH(p)}$. We can rewrite the binomial as follows:

$$\binom{n}{np} = 2^{\lg \binom{n}{np}}$$

So if we can prove that

$$\lg \frac{n!}{(n-np)!np!} = O(nH(p))$$

then we are done.

This is how the algebraic proof unfolds:

$$\begin{aligned}
\lg \frac{n!}{(n-np)!np!} &= \lg n! - \lg np! - \lg(n-np)! \\
&\leq c(n \lg n - np \lg np - (n-np) \lg(n-np)) \\
&\leq cn(\lg n - p \lg n - p \lg p \\
&\quad -(1-p) \lg n - (1-p) \lg(1-p)) \\
&\leq cn(-p \lg p - (1-p) \lg(1-p)) \\
&\leq cnH(p) \\
&= O(nH(p))
\end{aligned}$$

for some constant c, and noting that $\log n! \leq cn \log n$.

(c) We wish to prove that

Theorem 1

$$Pr\{D(E(x) \oplus \eta) = x\} < c^{-n}$$

for some constant c and every coding, decoding functions E, D, given $|x| = k$, every error vector η, and we know that $k/n > 1 - H(p)$.

Proof: First, we examine the probability that the error vector η is some fixed vector b. We know from part 4b that there are $\binom{n}{pn} = 2^{nH(p)}$ total error vectors. It follows then that

$$\Pr\{\eta = b\} \leq \frac{1}{\binom{n}{pn}} = 2^{-nH(p)+o(1)}$$

for big enough n.

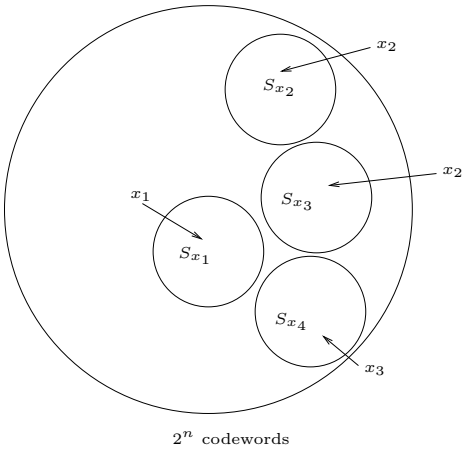

Figure 5: Every possible data word x_i is mapped to a subset of $\{0,1\}^n$ S_{x_i}

Next, we examine the space of all codewords $\{0,1\}^n$. We know that the length of a word of data before coding is k bits, so the total number of data words is 2^k. Also, through the Encoding function $E : 0,1^k \rightarrow 0,1^n$, we map every data word to a subset of the total codeword space S_x as seen in Figure 5

In this context, when $E(x_i) + \eta$ is a vector $y \in S_{x_i}$, we will be able to decode y to x, otherwise we will have an error. Therefore,

$$\Pr\{D(E(x) \oplus \eta) = x\} = \Pr_{x_i, y \in S_{x_i}}\{E(x_i) \oplus \eta = y\}$$

Note as well that for a given η, there is only one y that will be the result of $E(x_i) \oplus \eta$, so this is the same as writing $\eta = E(x_i) \oplus y$. Now we are ready to get the expression for

$$\begin{aligned}
\Pr\{D(E(x) \oplus \eta) = x\} &= \frac{1}{2^k} \sum_{x_i} \sum_{y \in S_{x_i}} \Pr\{(E(x) \oplus \eta) = y\} \\
&= \frac{1}{2^k} \sum_{x_i} \sum_{y \in S_{x_i}} \Pr\{\eta = E(x_i) \oplus y\} \\
&\leq \frac{1}{2^k} \sum_{x_i} \sum_{y \in S_{x_i}} 2^{-n(H(p)+o(1))} \\
&= \frac{2^n}{2^k} 2^{-n(H(p)+o(1))} \\
&= 2^{n-k} 2^{-n(H(p)+o(1))}
\end{aligned}$$

But remember that $k/n > 1 - H(p)$, and $n > k$ implying that $n - k = n(H(p) - \epsilon)$ for some $\epsilon > 0$:

$$\begin{aligned} \Pr\{D(E(x) \oplus \eta) = x\} &\leq 2^{n(H(p)-\epsilon)} 2^{-n(H(p)-o(1))} \\ &= 2^{-\epsilon n} \end{aligned}$$

■

5. **Fading and modulation**

   (a) Fading is the reduction in signal strength on the side of the receiver caused by multipath effects and mobility. Multipath effects cause several copies of the signal to arrive, disrupting the original desired signal. Multipath effects are:

   Reflection When the signal bounces off an obstacle

   Diffraction Edges in obstacles make the signal change its course, following the contour of the obstacle, causing a "bend" in the signal.

   Scattering When an obstacle of size comparable to the wavelength of the signal makes several fainter copies of the signal disperse around it

   Mobility also causes distortion in the received frequency in an amount proportional to the speed and the angle of the mobile node's path.

   To deal with fading, Adaptive Equalization (using a standard training sequence as a guideline to compensate for distortions), Forward Error Correction (adding redundant bits to a signal), and diversity techniques (sending the signal in multiple frequencies, at different times, multiple antennas, etc) are commonly used.

   (b) Amplitude modulation consists of changing the amplitude of the carrier signal to make it resemble the modulating signal. This is acheived by shifting the modulating signal and multiplying it by the carrier. In doing so, the amplitude of the carrier is a function of the original modulating signal. Mathematically, the transmitted signal $s(t)$ is built from the modulating $m(t)$ and carrier $c(t)$ as:

$$s(t) = (1 + m(t))c(t)$$

Taking $m(t), c(t)$ to be $M \cos 2\pi f_m t$ and $C \sin 2\pi f_c t$ respectively, we can obtain the following expression for $s(t)$:

$$s(t) = \sin(2\pi f_c t) + \frac{M}{2}\left(\sin 2\pi(f_c + f_m) + sin2\pi(f_c - f_m)\right)$$

Demodulation can be achieved simply by an envelope detector (a wave rectifier using a diode connected to a capacitor and a resistance) and a lowpass filter. (Seen on wikipedia)

(c) QAM combines two carriers that are out of phase with respect to each other by $\pi/2$. Two signals over QAM m_1, m_2 would yield a signal of the form:

$$s(t) = m_1(t)\cos(2\pi f t) + m_2(t)\sin(2\pi f t)$$

To demodulate the signal is multiplied by the sine carrier:

$$\begin{aligned}
\sin(2\pi f t)s(t) &= m_1(t)\sin(2\pi f t)\cos(2\pi f t) + m_2(t)\sin^2(2\pi f t) \\
&= \frac{1}{2}m_1(t)\sin(4\pi f t) + \frac{1}{2}m2(t)(1 - \cos(4\pi f t)) \\
&= \frac{1}{2}m_2(t) \\
&\quad + \frac{1}{2}(m_1(t)\sin(4\pi f t) - m_2(t)\cos(4\pi f t))
\end{aligned}$$

And and also by the cosine carrier:

$$\begin{aligned}
\cos(2\pi f t)s(t) &= m_1(t)\cos^2(2\pi f t) + m_2(t)\sin(2\pi f t)\cos(2\pi f t) \\
&= \frac{1}{2}m_1(t)(1 + \sin(4\pi f t)) \\
&\quad + \frac{1}{2}m_2(t)(4\pi f t) \\
&= \frac{1}{2}m_1(t) \\
&\quad + \frac{1}{2}(m_1(t)\sin(4\pi f t) + m_2(t)\cos(4\pi f t))
\end{aligned}$$

After passign both multiplied signal through a lowpass filter, $m2, m1$ can be recovered.