College of Computer Science
Northeastern University
Wireless Networks CS G250

Lecture 5
September 24 2008
Lecturer: R. Sundaram

Multiplexing, Spread Spectrum, Pseudo-Noise

Handouts: Simulation assignment (due on 10/06/08)

1 Multiplexing

Multiplexing is a term used to refer to a process where multiple analog message signals or digital data streams are combined into one signal over a shared medium. The aim is to share an expensive resource. There are two techniques used for multiplexing: time division multiple access(TDMA) and frequency divison multiple access(FDMA). The reverse process is known as demultiplexing. To multiplex a device named multiplexer (MUX) is used. The device that performs the reverse process is called a demultiplexer (DEMUX).

1.1 Time Division Multiple Access(TDMA)

It allows several users to share the same frequency channel by dividing the signal into different time slots. The users transmit one after the other each using his own time slot (Figure 1).

1.2 Frequency Division Multiple Access

In FDMA the available bandwidth is divided into frequency bands. Each transmitter is allocated a particular frequency band to send its data. The allocated frequency band belongs to the transmitter all the time. To prevent from interferences, the frequency bands are separated from each other by small guard bands(Figure 2).

2 Spread Spectrum

Spread spectrum techniques are used to transmit either analog or digital data, using an analog signal. The idea is to spread the information signal

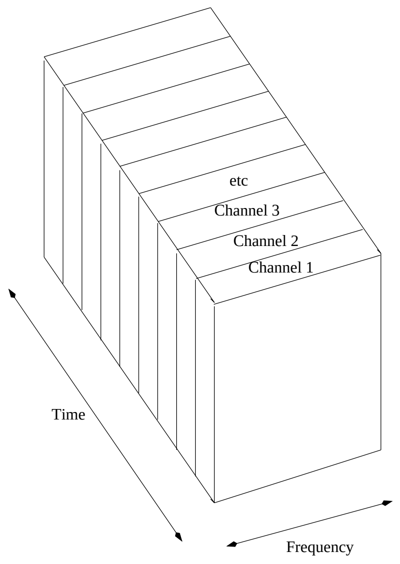

Figure 1: TDMA

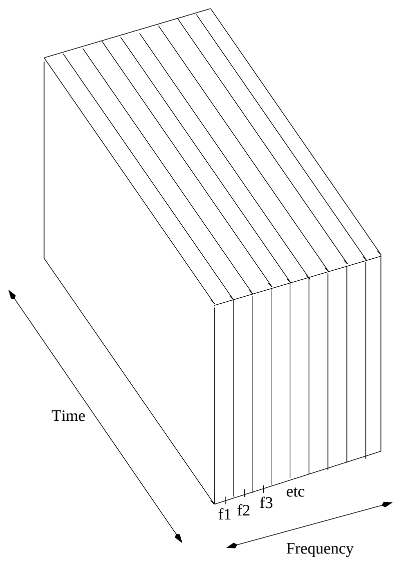

Figure 2: FDMA

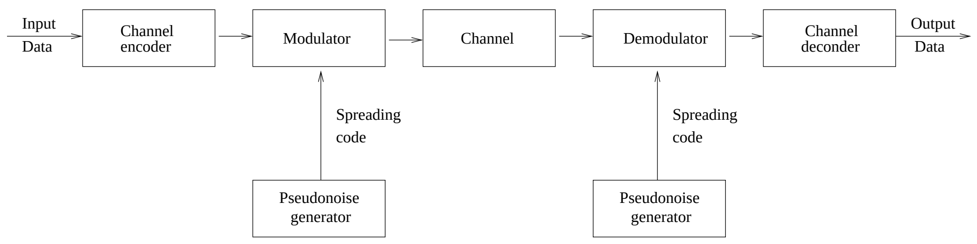

Figure 3: General Model of Spread Spectrum Communication Systems

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.

2.1 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 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.

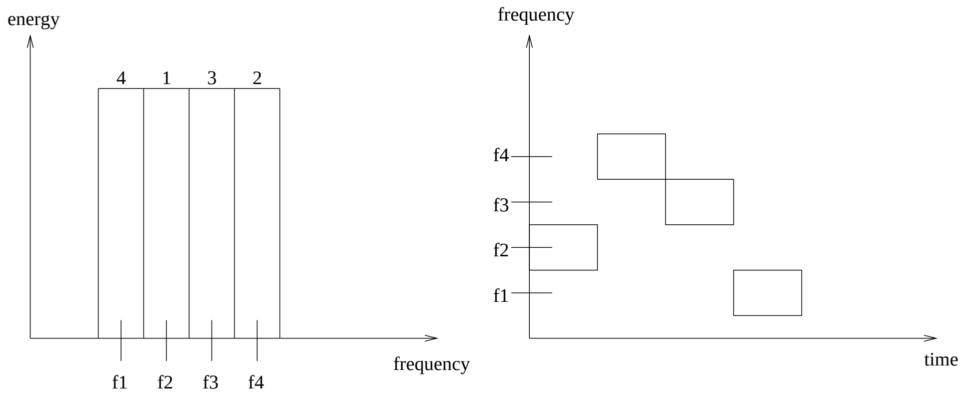

Figure 4: Frequency Hopping Spread Sprectrum - Example

2.1.1 FHSS using BFSK (Binary Frequency Shift Keying)

For transmission, binary data are fed into a modulator using some digital-to-analog encoding scheme, such as BFSK. A pseudonoise serves as spreading code. Mathematically,

$$\begin{aligned}
s_d(t) &= A\cos\left(2\pi\left(f_0 + 0.5\left(b+1\right)\Delta f\right)t\right), \text{where} \\
A &= \text{amplitude} \\
f_0 &= \text{base frequency} \\
\Delta f &= \text{frequency separator} \\
b &= \text{value of the bit (1 for binary 1; -1 for binary 0)}
\end{aligned}$$

Why 0.5? Because when $b = -1$, then $s_d(t) = A\cos(2\pi f_0 t)$ and when $b = 1$, then $s_d(t) = A\cos(2\pi(f_0 + \Delta f)t)1$. Thus the frequency of the data signal is either f_0 for $b = -1$ or $f_0 + \Delta f$ for $b = 1$.

To this signal $s_d(t)$ we apply a chipping signal $c(t)$ which hops to a different frequency among a set of 2^k frequencies after k bits from the sequence. For simplicity reasons, let's assume we can send only one bit during one frequency hop. Then the signal during the i-th hop is:

$$\begin{aligned} p(t) &= s_d(t)\,c(t)\,, \text{where} \\ c(t) &= \text{chipping sequence} \\ c(t) &= \cos(2\pi f_i t) \\ \text{Thus} & \\ p(t) &= A\cos(2\pi(f_0 + 0.5(b+1)\Delta f)\,t)\cos(2\pi f_i t) \end{aligned}$$

By applying the following trigonometric identity

$$\cos x \cos y = \frac{1}{2}(\cos(x+y) + \cos(x-y))\,,$$

we get

$$\begin{aligned} p(t) &= 0.5A\,[\cos(2\pi(f_0 + 0.5(b_i+1)\Delta f + f_i)\,t) \\ &\quad + \cos(2\pi(f_0 + 0.5(b_i+1)\Delta f - f_i)\,t)] \end{aligned}$$

A bandpass filter is used to block the difference frequency and pass the sum frequency, yielding an FHSS signal of

$$s(t) = 0.5A\cos(2\pi(f_0 + 0.5(b_i+1)\Delta f + f_i)\,t)$$

At the receiver, a signal of the form $s(t)$ will be received. This is multiplied by a replica of the spreading signal to yield a product signal of the form:

$$\begin{aligned} p(t) &= s(t)\,c(t) \\ &= 0.5A\cos(2\pi(f_0 + 0.5(b_i+1)\Delta f + f_i)\,t)\cos(2\pi f_i t) \end{aligned}$$

Using the trigonometric identity, we get:

$$\begin{aligned} p(t) &= s(t)\,c(t) \\ &= 0.25A\,[\cos(2\pi(f_0 + 0.5(b_i+1)\Delta f + f_i + f_i)\,t) \\ &\quad + \cos(2\pi(f_0 + 0.5(b_i+1)\Delta f)\,t)] \end{aligned}$$

A bandpass filter is used to block the sum frequency and pass the difference frequency, yielding the signal $s_d(t) = \cos(2\pi(f_0 + 0.5(b_i + 1)\Delta f)t)$.

2.1.2 FHSS using MFSK (Multiple Frequency Shift Keying)

MFSK uses $M = 2^L$ different frequencies to encode the digital input L bits at a time. The transmitted signal is of the form:

$$\begin{aligned}
s_i(t) &= A\cos 2\pi f_i t, \text{where } 1 \leq i \leq M \\
\text{and} & \\
f_i &= f_c + (2i - 1 - M)f_d, \text{where} \\
f_c &= \text{carrier frequency} \\
f_d &= \text{difference frequency} \\
M &= \text{number of different signal elements} \\
M &= 2^L \\
L &= \text{number of bits per signal element}
\end{aligned}$$

Let T_s be the duration of a signal element and T_c the number of seconds when I am sending on a certain frequency.

According to the relationship between T_c and T_s we can characterize the spread spectrum as:

- slow-frequency-hop spread spectrum $T_c \geq T_s$
  In this situation we will be able to pack more blocks of size L.

- fast-frequency-hop spread spectrum $T_c < T_s$
  In this situation, we are unable of sending the entire block of size L. This is extremely useful in military applications.

The gain in signal-to-noise ratio or processing gain is:

$$\begin{aligned}
G_P &= \frac{W_s}{W_d} \\
&= 2^k \\
W_d &= \text{the bandwidth of the transmitter} \\
W_s &= \text{the bandwidth of a slot}
\end{aligned}$$

So, if frequency hopping is used the jammer must jam all 2^k frequencies (Figure 5).

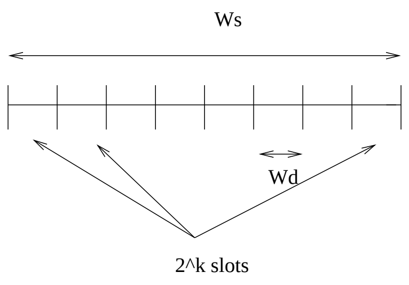

Figure 5: Signal-to-noise ratio

2.2 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.

2.2.1 DSSS using BPSK

Rather than representing binary data with 1 and 0, it is easier for the current purpose to use +1 and -1 to represent the two binary digits. In this case, a BPSK signal can be represented as:

$$\begin{aligned} s_d(t) &= Ad(t)\cos(2\pi f_c t), \text{where} \\ A &= \text{the amplitude of the signal} \\ f_c &= \text{carrier frequency} \\ d(t) &= \text{the discrete function that takes the value +1 for 1 and -1 for 0} \end{aligned}$$

On the transmitter's side, the signal is multiplied by $c(t)$, which is the pseudo-noise sequence.

$$\begin{aligned} s(t) &= s_d(t)\,c(t) \\ &= Ad(t)\,c(t)\cos(2\pi f_c t) \end{aligned}$$

On the receiver's side, the incoming signal is multiplied again by $c(t)$.

$$\begin{aligned} s(t)\,c(t) &= Ad(t)\,c(t)\,c(t)\cos(2\pi f_c t) \\ c(t)\,c(t) &= 1 \\ \text{Therefore,} & \\ s(t)\,c(t) &= s_d(t) \end{aligned}$$

2.3 Code Division Multiple Access (CDMA)

This technique allows the transmitters to send the information simultaneously over a single communication channel. Each transmitter is assigned a code which allows multiple users to be multiplexed over the same physical channel. The codes assigned to the users have the nice property of being orthogonal. Two vectors are orthogonal if their inner product is 0.

Let's suppose we have two users: A and B. User A has been assigned $C_A = \langle 1, -1, 1, -1 \rangle$ and user B has been assigned code $C_B = \langle 1, -1, -1, 1 \rangle$. If user A wants to send a 1, then he sends $C_A = \langle 1, -1, 1, -1 \rangle$. If the same user A wants to send a 0, then he sends $-C_A = \langle -1, 1, -1, 1 \rangle$, which is obtained by flipping each value of code C_A. The same applies to user B.

To decode, the receiver applies the following formula: $\frac{1}{n} s {C_A}^T$, where n is the number of chips and s is the received code. So, in our example, if $s = \langle 1, -1, 1, -1 \rangle$ and $n = 4$, then the received value is:

$$\frac{1}{4} \langle 1, -1, 1, -1 \rangle \langle 1, -1, 1, -1 \rangle^T = \frac{4}{4} = 1,$$

meaning that user A sent a 1. If $s = \langle -1, 1, -1, 1 \rangle$ and $n = 4$, then the received value is:

$$\frac{1}{4} \langle -1, 1, -1, 1 \rangle \langle 1, -1, 1, -1 \rangle^T = \frac{-4}{4} = -1,$$

meaning that user A sent a 0.

So, what happens if we receive a $s = \langle 1, -1, -1, 1 \rangle$ and we apply C_A? By applying the above formula,

$$\frac{1}{4} \langle 1, -1, -1, 1 \rangle < 1, -1, 1, -1 >^T = \frac{0}{4} = 0.$$

So, if we apply the right code we get either 1 or -1, but if we apply the wrong code we always get 0.

On the receiver's side, it could happen that we get the sum of the codes sent by A and B: $r = \left(s_A \bar{C_A} + s_B \bar{C_B}\right)$. If we want to find out the signal sent by A, we multiply r by ${C_A}^T$. Because C_B and C_A are orthogonal, $\left(s_A\bar{C_A} + s_B\bar{C_B}\right){C_A}^T = s_A$. A similar thing happens when we want to extract the signal sent by B.

Question: If we have 2^k channels, how many pairs of vectors can we have so that they are all orthogonal to each other?

Answer: 2^k, meaning that this is the maximum number of people that can send data simultanously.

Orthogonal codes are really nice to have, but the problem with them is that there aren't that many.

3 Pseudo-Noise

Pseudo-noise has to satisfy two properties: randomness and unpredictability. Pure randomness is hard to achieve. Therefore we need to settle for pseudo-randomness, which means that the sequences produced are long and there is no way of predicting the next number from the sequence.

One way of generating pseudo-noise is by using Linear Feedback Shift Registers(LFSR). LFSR are implemented as a circuit consisting of XOR gates and shift register. The register is a string of 1-bit storage devices. The main idea is that the k-th bit is output as the result of an operation on the previous $k-1$ bits in the register. By using a long enough sequence of bits, the sequence may appear to be random, even though it is actually a long cycle.