CHAPTER 2

The Wireless Link

This chapter uses the example of communication between two users to informally introduce some concepts related to the physical layer. These concepts will be useful in understanding wireless protocol design and performance.

2.1 An Example Scenario

Let us consider two users, Mary and John, who communicate with each other using a voice-over-IP application, as shown in Figure 2.1. Mary and John are both connected to a computer, with the link connecting the two computers being a wireless link.

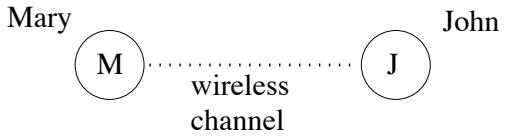

Figure 2.1 Voice-over-IP over a wireless link

This chapter will discuss how Mary's "signal" – that is, her speech, is delivered to John's ears. We will divide the "lifespan" of the signal in three phases: (I) processing at the transmitter, (II) propagation through the wireless channel, and (III) processing at the receiver. For this discussion, it will also be useful to refer to Figure 2.2, which illustrates the different functional blocks at the transmitter and the receiver.

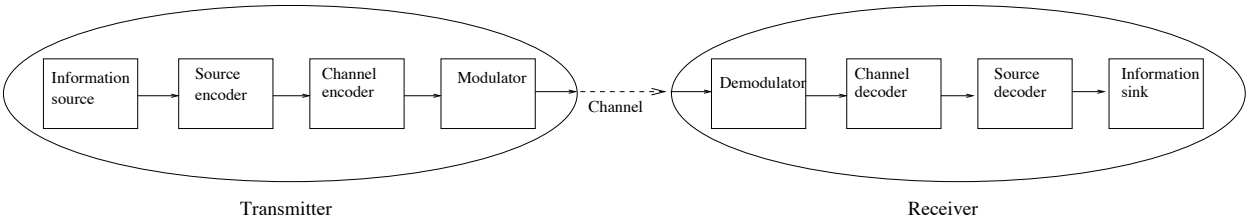

Figure 2.2 A digital communication link

2.2 Processing at the Transmitter

At the transmitter, the speech produced by Mary must be somehow translated into an electrical signal, in a form suitable for transmission on the wireless channel. Several steps are typically used to achieve this goal, as discussed in this section.

2.2.1 Digital Representation of Information

Mary is speaking into a microphone attached to her computer. The words spoken by Mary create an analog audio signal, which is translated into analog electrical signal by the microphone. This signal can then be sampled, and the resulting sampled values can be represented in binary. The sampled data may also be compressed so that the spoken words can be represented using fewer bits. For instance, when Mary pauses every once in while, the resulting silent intervals may be "compressed" to reduce the number of bits necessary. In general, rather than bits, the signal may be represented using larger *symbols*, each symbol corresponding to several bits. In our discussion, we will assume that each symbol corresponds to a single bit.

2.2.2 Packetization

When Mary speaks into the microphone, a sequence of bits is generated as seen above. Since Mary and John are having a conversation, we would like John to be able to quickly start hearing Mary's response to whatever he may have said. Because of this, we cannot wait for Mary to complete her response – which may very well last several minutes! – before we start sending the corresponding bits to John. To achieve this goal, the bits obtained after source coding are divided into chunks of consecutive bits. We will refer to this process as *packetization*. For instance, we may decide to form one chunk corresponding to each 20 ms duration of Mary's speech. This implies that 50 packets may be formed for each second for which Mary talks.

2.2.3 Encapsulation

With packetization, we obtain a series of packets that contain the bits representing Mary's voice. Now these packets must be sent to John's computer. We know that the internet protocol (IP) may not deliver the packets in the correct order to John's computer. Also, some packets may be delivered very late, and would be useless in "playing back" Mary's voice to John. To allow identification of such late packets on John's computer, it is necessary to include additional information in each packet. This can be accomplished by attaching an *application layer* header to the packets obtained above, and including in this header the information that would be useful to the voice-over-IP application on John's computer in reproducing Mary's voice correctly (such as a timestamp).

Having formed this application layer packet, we would now need to deliver it to John's computer. On John's computer, many applications may be running simultaneously, and we need some mechanism to ensure that Mary's voice packets are delivered to the right application. This goal can be accomplished using a *transport protocol*. For instance, we can use User Datagram Protocol (UDP) for this purpose. By making use of the *IP address* for John's computer, and the *UDP port* used by John's voice-over-IP application, UDP can ensure that the packets are delivered to the right application. For this purpose, the packets obtained above (including the application layer header) are *encapsulated* by attaching a UDP header to the packet, to obtain a UDP segment.

The UDP protocol, which resides at the transport layer of the protocol stack, then passes the segment to the Internet Protocol (IP). An IP header is attached to the above segment, to obtain an IP datagram. Information in the IP header is useful in performing routing.

IP then passes the datagram down to the link layer. If "Wi-Fi" or IEEE 802.11 protocol is used as the medium access control (MAC) protocol on the wireless link between the two computers, then a IEEE 802.11 MAC header is added to the datagram to create an IEEE 802.11 frame. The frame is then passed to the physical layer for transmission on the wireless channel.

2.2.4 Error Control Codes (ECC)

When a signal is transmitted on the wireless channel, the received signal is not identical to the transmitted signal. The signal "deteriorates" during propagation from the transmitter to the receive, and also affected by interference from other transmitters. Therefore, sometimes the receiver has difficulty correctly determining the bits that were transmitted. Thus, *errors* may occur, which cause a transmitted 0 bit to be received as a 1 and vice-versa. To allow the receiver to detect and/or correct such errors, the *channel coding* module

at the physical layer introduces redundancy in the transmitted data. For instance, the IEEE 802.11a physical layer uses a *convolutional code* to allow correction of some errors.

In practice, error control codes are also used at other layers of the protocol stack, introducing redundancy at several layers. For instance, the TCP protocol uses a checksum to detect errors in data and the TCP header. The TCP checksum is computed before passing the packet to IP. When IP receives the packet from a higher layer protocol such as TCP or UDP, it adds the IP header along with a checksum for the IP header. Suppose that IP hands the datagram to the MAC layer for IEEE 802.11a. The IEEE 802.11 MAC header includes a cyclic redundancy check, which can be used to detect some errors in a MAC layer frame. As mentioned above, the IEEE 802.11a physical layer uses a convolutional code when transmitting the MAC layer frame. At the receiver, all of these codes will be used suitably at the various layers of the protocol stack, as the packet travels up the stack from the physical layer.

Appendix A provides further discussion of error-control codes, including some simple examples.

2.2.5 Modulation

Modulation is the process by which the packet to be transmitted (which is a sequence of bits) is encoded on a "carrier". The carrier frequency is chosen depending on the channel characteristics, as well as policy restrictions. Each channel can efficiently carry information at certain frequencies. Also, there are often policy restrictions on the frequencies that may be used for a certain communication. The process of modulation allows us to use desired frequency range for the communication.

The carrier frequency is normally chosen to be much larger than the bandwidth of the signal to be transmitted. We will define the term *bandwidth* more precisely later in this section, but intuitively, bandwidth is a measure of the amount of spectrum used by a signal. If B is the signal bandwidth, and f_c is the carrier frequency, then let us assume that $f_c >> B$. For instance, the wireless channel may be assigned $B = 20$ MHz bandwidth with $f_c = 915$ MHz. In this section, we will discuss the process of modulation that occurs at a transmitter. Modulation can be performed in many different ways. As an example, we will discuss one of the simpler modulation schemes, namely, binary pulse amplitude modulation (binary PAM).

In case of binary PAM, the information is transmitted one bit at a time, that is, with one bit per symbol. Alternatively, we can group several bits into a single symbol, and encode them together on the carrier. For instance. QPSK (Quadrature Phase-Shift Keying) encodes 2 bits per symbols. While the information in the symbols is digital, the signal

produced as a result of the modulation process is an analog signal. This analog signal is transmitted on the wireless channel.

Binary PAM can be divided into two steps [19]:

- Represent each information bit input using a "baseband" signal: We will use a suitably designed pulse $b(t)$, such that $b(t)$ takes non-zero values only if $0 \leq t \leq T$ for some T. One pulse will be transmitted each T time units.

  With binary PAM, each symbol can take values 0 or 1, and a pulse corresponding to one symbol is transmitted for each T duration. Thus, j-th pulse ($j \geq 1$) is transmitted during time interval from $(j-1)T$ to jT. In our discussion below, let us focus on the very first symbol transmitted, with the corresponding pulse occupying time interval $[0, T]$. We encode values 0 and 1 for the first symbol as waveform $s_0(t)$ and $s_1(t)$, respectively, as defined below:

$$\begin{aligned} s_0(t) &= -b(t) \\ s_1(t) &= b(t) \end{aligned}$$

  In general, to transmit 0 as the j-th symbol, the signal waveform will be given by $-b(t-(j-1)T)$, and to transmit 1 as the j-th symbol, the waveform will be given by $b(t-(j-1)T)$.

  The above process results in the baseband signal corresponding to the sequence of bits to be transmitted. The baseband signal is thus a sum of time-shifted copies of $b(t)$ and $-b(t)$.

- Superimpose the signal on the carrier: This step "shifts" the baseband signal to a higher carrier frequency. Let f_c denote the carrier frequency. For pulse amplitude modulation, modulation is achieved simply by multiplying the signal by the "carrier". Here we focus our attention on just the first symbol transmitted at time 0. In particular, to transmit value i (i is 0 or 1) at time 0, we transmit the waveform

$$\begin{aligned} w_i(t) &= s_i(t)\ \cos(2\pi f_c t) \\ &= A_i\, b(t)\ \cos(2\pi f_c t) \end{aligned}$$

  where $A_i = -1$ for $i = 0$ and $A_i = 1$ for $i = 1$.

Frequency Spectrum after Modulation: To intuitively understand why multiplication by the carrier $\cos(2\pi f_c t)$ results in a shift in frequency spectrum, let us consider multiplying a baseband signal $\cos(2\pi f t)$ by $\cos(2\pi f_c t)$, where $f << f_c$. Then we have

$$\cos(2\pi f t)\ \cos(2\pi f_c t) = \frac{1}{2}\cos(2\pi(f_c - f)t)\ +\ \frac{1}{2}\cos(2\pi(f_c + f)t)$$

Thus, while the unmodulated signal is at frequency f, the modulated signal consists of sinusoids at frequencies $f_c - f$ and $f_c + f$. In general, above modulation process results in a transmitted signal with the spectrum centered at the carrier frequency. This phenomenon is easier to discuss using the Fourier transform for the signal. Fourier transform of a signal is a way of representing the manner in which the signal "occupies" the frequency spectrum. Appendix B provides some intuition behind the definition of Fourier transform. Specifically, if $X(f)$ is the Fourier transform of a signal $x(t)$, then $x(t)$ can be obtained using the following inverse Fourier transform operation:

$$x(t) = \int_{-\infty}^{+\infty} X(f) e^{j2\pi ft}\, df$$

f in $X(f)$ denotes frequency, and t in $x(t)$ denotes time. Thus, $X(f)$ is the frequency-domain representation of a signal, and $x(t)$ is the time-domain representation of the same signal. In general, Fourier transform $X(f)$ may be a complex number. f may be positive or negative. It is easy to show that for real-valued signal $x(t)$, the magnitudes of $X(f)$ and $X(-f)$ are identical. A real-valued signal has bandwidth W if, for $f \geq 0$, the Fourier transform is non-zero only in a frequency band of width W. Since $|X(f)| = |X(-f)|$, it follows that, for $f \leq 0$ as well, the Fourier transform for the real-valued signal is non-zero only in a band of width W. This is illustrated in Figure 2.3(a).

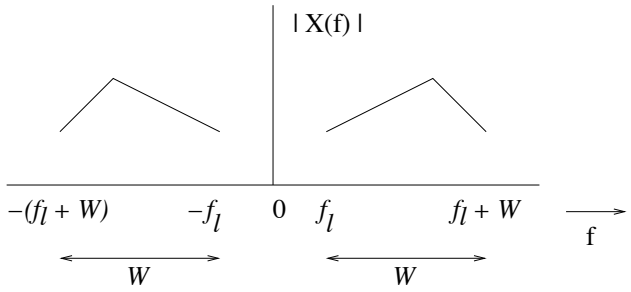

Figure 2.3 Bandwidth W of a real signal

It turns out that if $X(f)$ is the Fourier transform of some signal $x(t)$, then the Fourier transform of signal $x(t)cost(2\pi f_c t)$ is given by

$$\frac{1}{2}X(f - f_c) + \frac{1}{2}X(f + f_c)$$

It follows that if $x(t)$ is a baseband signal band-limited to $[-B/2, B/2]$, the modulated signal will be band-limited to frequencies f such that $f_c - B/2 \leq |f| \leq f_c + B/2$. Thus, the modulated signal will have bandwidth B centered at f_c. This explains how modulation can be used to "shift" a baseband signal to the desired center frequency. In case of binary PAM, the shape of the pulse $b(t)$ will determine the spectrum occupied by the modulated signal.

The modulated signal is then transmitted on the wireless channel from host M to host J in our example in Figure 2.1. This signal propagates through the wireless channel to the receiver. Processing is performed at the receiver to recover the transmitted signal. In Section 2.3, we discuss propagation of the signal through the wireless channel, and processing at the receiver is discussed in Section 2.4.

Energy and Power Content of a Signal: To aid in the discussion later in this chapter, we now introduce the notion of *energy content* and *power content* of a signal. The energy content of signal $x(t)$ over duration $[t_1, t_2]$ is defined as

$$E_x = \int_{t_2}^{t_1} |x(t)|^2 \, dt \tag{2.1}$$

Power content P_x of a signal $x(t)$ is defined as

$$P_x = \lim_{T \to \infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} |x(t)|^2 \, dt \tag{2.2}$$

Circuit theory tells us that voltage or current $x(t)$ applied across a 1 Ω resistance will result in power dissipation of $x^2(t)$. Thus, P_x may be interpreted as the average amount of energy that will be dissipated by a 1 Ω resistor in 1 second if $x(t)$ is the voltage or current applied to the resistor.

Previously we have seen that a time-domain signal can also be represented in the frequency-domain using its Fourier transform. Similarly, while we defined power content above using the time domain, it is also possible to define a frequency-domain function – namely, *power spectral density* – to obtain power content of a signal. In particular, if we denote power spectral density of signal $x(t)$ as $S_x(f)$, then power content of $x(t)$ can be obtained as

$$P_x = \int_{-\infty}^{+\infty} S_x(f) \, df \tag{2.3}$$

Power spectral density $S_x(f)$ may be interpreted as a measure of the contribution of frequency f to the power content of signal $x(t)$. The above notion of power spectral density is useful for deterministic signals for which the signal value at each time t is deterministically chosen, as well as for non-deterministic signals (such as noise) for which the signal value at time t is a random variable. For brevity, we will not discuss how to calculate the power spectral density here. For future reference, note that when a signal has a finite bandwidth, the power spectral density for the signal is non-zero only over the signal bandwidth.

Power content of a signal is measured in Watts (W). Decibel notations dBW and dBm are often used to represent the power level. Power P in Watts can be converted to the decibel units dBW using the formula $10 \log_{10} P$. Similarly, power P in milliwatts can be converted to decibel units dBm using the formula $10 \log_{10} P$. In our discussion below, the base of logarithms used in determining decibel quantities will be assumed to be 10, even if the base is not specified explicitly.

2.3 The Wireless Channel

The signal transmitted by the transmitter is eventually received by the receiver. The received signal is different from the transmitted signal for three main reasons:

- *Channel characteristics*: When the signal propagates from the transmitter to the receiver, the signal attenuates as the distance increases. Also, the environment through which the signal propagates includes objects that may also modify the signal. For instance, if the signal propagates through walls, the materials in the walls will attenuate the signal more than propagation through free space. Also, signal may be reflected or scattered by the the objects in the environment, causing the signal to reach the receiver along many different paths.
- *Noise:* Thermal noise is introduced by the hardware used for communication. Thus, the receiver receives a composite of the signal and the noise.
- *Interference:* Simultaneous transmissions can pose interference to each other. Such an interferer is not shown in Figure 2.1, although interference is commonplace in most wireless environments.

The impact of the channel, noise, and interference can be characterized using a simple equation as follows. Let $r(t)$ be the received signal at receiver J at time t, and let $x(t)$ be the signal transmitted by host M at time t. Also, let $n(t)$ denote the noise at time t, and let $i(t)$ be interfering signal at the receiver J at the time t. Note that $i(t)$ is the composite of the interference from all the interference sources, which are not shown in Figure 2.1. Assuming that the signal travels from the transmitter to the receiver along different paths, with path i having a delay of τ_i and attenuation factor a_i, the received signal can be obtained as follows:

$$r(t) = \sum_i a_i \, x(t - \tau_i) \; + \; n(t) \; + \; i(t) \tag{2.4}$$

where the summation is over all the paths taken by the signal when propagating from the transmitter to the receiver. In general, the channel conditions can be time-varying, making a_i and τ_i functions of time.

The noise $n(t)$ in the above equation is typically modeled using an Additive White Gaussian Noise (AWGN) process. Each of the words *Additive, White, Gaussian* and *process* deserves some explanation. The term *additive* refers to the fact that noise *adds* to the signal, as seen in the above equation. The term *process* implies that the noise $n(t)$ at each time t is a random variable. For the AWGN process, the noise at different instants of time is independent of each other. The term *Gaussian* represents that $n(t)$ has a Gaussian distribution with zero mean. It is customary to denote the variance of this Gaussian distribution as $\frac{N_0}{2}$.

If $\mathcal{N}$ denotes the Gaussian random variable for additive white Gaussian noise at a certain time, then

$$p(\mathcal{N} = n) = \frac{1}{\sqrt{\pi N_0}}\, e^{-\frac{n^2}{N_0}} \tag{2.5}$$

$p(\mathcal{N} = n)$ above is the probability that the actual additive noise amplitude at time t is equal to n.

The term *white* in AWGN implies that the power spectral density for the noise is "flat" or constant for all frequencies. Thus, the noise power only depends on the bandwidth, not which part of the spectrum is utilized. In fact, it turns out that the noise power spectral density for the AWGN channel is equal to the variance $\frac{N_0}{2}$ of the Gaussian random variable in Equation 2.5. Thus, if the bandwidth for the signal of interest is W, then the noise introduced over that bandwidth will have power $N = \frac{N_0}{2}(2W) = N_0 W$ Watts. Recall that by our definition of bandwidth for real signals, a signal with bandwidth W occupies positive frequencies over a band of width W, and also negative frequencies over a band of width W, for a total of $2W$. Hence the noise power is given by $\frac{N_0}{2}(2W)$. For instance, when W is 10 MHz and $N_0/2$ is 4×10^{-21} W/Hz, we have noise power equal to 8×10^{-14} Watts or -131 dBW or -101 dBm.

AWGN model is an approximation of the reality: since the total amount of spectrum is infinite, the total noise power over the entire spectrum will be ∞! Obviously, we don't have noise with infinite power content. However, in practice, we use only a small fraction of the available spectrum, and over the bandwidths of interest, the approximation of "flat" power spectral density is adequate.

2.4 Processing at the Receiver

When the transmitted signal propagates to the receiver, the receiver must perform several steps to recover the transmitted information. As seen earlier, the signal obtained after the modulation step is transmitted on the wireless channel. At the receiver, *demodulation* is performed, to attempt to recover the transmitted symbols. As an illustration, this section discusses demodulation of binary PAM signals. We will make several simplifying assumptions in our discussion:

- Let us assume that there is only a line-of-sight path between the transmitter and the receiver with delay τ.
- Equation 2.4 tells us how the received signal can be related to the transmitted signal. Let us assume that there is no interference (that is, $i(t) = 0$), or equivalently, that the

interference appears similar to noise, and therefore, the interference can be absorbed into the $n(t)$ term in Equation 2.4.

This assumption, together with the assumption of a single line-of-sight path implies that, the received signal can be written as follows, where a is a constant, and $x(t)$ is the transmitted signal:

$$r(t) = a\,x(t-\tau) \;+\; n(t) \tag{2.6}$$

- Let us assume that the transmitter and the receiver are perfectly synchronized, and the receiver knows when the bit boundaries occur (such a demodulator is said to be a *coherent* demodulator). Under these conditions, intuitively, we can run the clock at the receiver behind the clock at the transmitter by τ, allowing us to rewrite the equation for $r(t)$ as follows:

$$r(t) = a\,x(t) \;+\; n(t) \tag{2.7}$$

  In essence, we can ignore the delay τ.

- In practical systems, the pulses transmitted for adjacent bits may potentially overlap, leading to *intersymbol interference* (ISI). We will make the simplifying assumption that there is no inter-symbol interference (ISI). This assumption allows us to analyze the demodulation for a bit without having to take into account ISI.

Let us consider demodulation of the received signal corresponding to the very first bit received by the receiver. Each subsequent bit can be demodulated similarly. The first step in the demodulation is to remove the carrier. Recall that the transmitted signal for bit value i ($i = 0$ or 1) is $w_i(t)$. Thus, from equation 2.7, the waveform received at the receiver is given by $r(t) = a\,w_i(t) + n(t)$, where $n(t)$ is the noise. Assume the AWGN model, with variance of $n(t)$ being $N_0/2$.

Let us denote E_b as the energy content in $a\,w_i(t)$ – which is the signal received at the receiver ignoring noise – over bit duration $[0, T]$. This can be viewed as the energy-per-bit received from the transmitter. Then,

$$\begin{aligned} E_b &= \int_0^T (a\,w_i(t))^2 dt \\ &= a^2 \int_0^T b^2(t)\cos^2(2\pi f_c t) dt \end{aligned}$$

To remove the carrier, the demodulator performs the following operation:

$$z = \frac{1}{\sqrt{E_b}} \int_0^T r(t)\,a\,b(t)\cos(2\pi f_c t)\,dt$$

To perform this operation, the demodulator will need to know the value of a (using which it can also computer E_b). Thus, we are implicitly assuming that the receiver somehow knows

a. This assumption is not quite necessary, but makes the analysis more appealing. As you can see soon, removing $\frac{a}{\sqrt{E_b}}$ from the above expression will simply result in a linear scaling of z. The above expression can be rewritten as

$$\begin{aligned} z = \frac{1}{\sqrt{E_b}} \int_0^T r(t)\, a\, b(t) \cos(2\pi f_c t)\, dt &= \frac{1}{\sqrt{E_b}} \int_0^T a\, w_i(t)\ [a\, b(t) \cos(2\pi f_c t)]\ dt\ + \\ & \frac{1}{\sqrt{E_b}} \int_0^T n(t)\ [a\, b(t) \cos(2\pi f_c t)]\ dt \end{aligned} \tag{2.8}$$

Now, let us consider the first integral on the right hand side of Equation 2.8.

$$\begin{aligned} \frac{1}{\sqrt{E_b}} \int_0^T a\, w_i(t)\, a\, b(t) \cos(2\pi f_c t)\, dt &= \frac{1}{\sqrt{E_b}}\, a^2 \int_0^T A_i\, b^2(t) \cos^2(2\pi f_c t) dt \\ &= \frac{1}{\sqrt{E_b}}\, A_i\, E_b \\ &= A_i \sqrt{E_b} \end{aligned}$$

Now, let us consider the second integral on the right hand side of Equation 2.8, and denote its value by m. Thus,

$$m = \frac{1}{\sqrt{E_b}} \int_0^T n(t)\, a\, b(t) \cos(2\pi f_c t)\, dt$$

Since $n(t)$'s are independent Gaussian random variables with mean 0, the above integral can be viewed as a linear combination of independent zero-mean Gaussian random variables, namely, $n(t)$, $0 \le t \le T$. Therefore, it follows that m is also a Gaussian random variable with mean 0. It can also be shown that m has variance $N_0/2$.

The above discussion implies that the output of the demodulator is given by

$$z = A_i \sqrt{E_b} + m$$

where m is a Gaussian random variable with mean 0 and variance $N_0/2$. In the absence of noise (or, with $m = 0$), the output will be $-\sqrt{E_b}$ if the transmitted bit is 0, and $+\sqrt{E_b}$ if the transmitted bit is 1. Now, suppose that the transmitter transmits bits 0 and 1 with equal probability. Then, since m is a random variable symmetric around 0, it follows that the optimal decision rule to decide whether the transmitter has sent 0 or 1 is as follows:

- 0 if $z < 0$
- 1 if $z > 0$
- 0 or 1 (arbitrary choice) when $z = 0$

The received bit will be erroneous, if the transmitter sends 1 but the demodulator output is 0, and vice-versa. In other words, the decision at the receiver will lead to an error if $m > \sqrt{E_b}$ when the transmitted bit is 0, and $m < -\sqrt{E_b}$ when the transmitted bit is 1. Since m is a Gaussian variable with variance $N_0/2$, the bit error probability P_b is then given by

$$\begin{aligned} P_b = \int_{\sqrt{E_b}}^{\infty} \frac{1}{\sqrt{\pi N_0}} e^{-\frac{u^2}{N_0}} du &= Q\left(\frac{\sqrt{E_b}}{\sqrt{N_0/2}}\right) \\ &= Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \end{aligned}$$

where $Q(v)$ is defined as probability that a Gaussian variable with mean 0 and variance 1 is greater than v, that is, $Q(v) = \int_v^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} du$.

Observe that the error probability is a decreasing function of E_b, which is the energy-per-bit in the received signal. If we denote received power as P_r and the bit rate as R, then it follows that $E_b = P_r/R$. Thus, the error probability is a function of P_r/N_0, or the signal-to-noise ratio (SNR). In general, the reception may also encounter interference from other sources, and the presence of interference may increase the error probability.

The receiver can demodulate each bit in a packet and form the *received packet*. Now, as seen above, some of the demodulated bits may be in error. Such errors may be detected (and possibly corrected) using error control codes used in the channel coding stage. After error correction at the physical layer, the packet is passed to the link layer for processing. The link layer may apply its own error checks to the received packet and in absence of any detected errors, pass the packet to the next layer. Similarly, IP will verify the checksum for the IP header, and discard the packet if an error is detected. Why do we need such checks at multiple layers of the protocol stack? First, the error control code at a lower layer cannot correctly fix all possible errors in a packet. Thus, checks at higher layers can potentially detect errors that occur despite the use of codes at the lower layer(s). Also, errors may potentially be introduced at the lower layers due to software bugs, or memory corruption. Higher layer checks can help detect such errors as well.

In our example, when a packet eventually reaches the application layer on John's computer, the application layer will translate the received samples of Mary's voice into an audible signal using a speaker, which can then be heard by John.

2.5 Throughput Limit for a Wireless Link

Performance of a traffic flow can be measured using various parameters, such as throughput, packet loss rate, delay, and jitter. For the voice-over-IP flow in our example, all of these parameters may of interest. For a bulk data transfer, we may be more interested in just the throughput, that is, the rate at which data is delivered reliably between the two hosts.

Shannon introduced the notion of *capacity* to characterize the best achievable rate of reliable information delivery. Reliable communication is feasible only at rates that do not exceed the capacity. While determining capacity is a difficult problem in general, in some specific cases, precise formulations of capacity are known. In this section, we summarize one such result, which will be used later in the book.

Consider a wireless link between a pair of hosts as in Figure 2.1. Suppose that the channel between these hosts is a AWGN channel with noise power spectral density $N_0/2$. The capacity C of such a link is given by

$$C = W \log_2 \left(1 + \frac{P}{N_0 W}\right) \qquad \text{bits/second}$$

where P is the received signal power (at the receiver), and W is the channel bandwidth (in Hz), assuming interference $I = 0$.

In general, there may also be interference from other transmitters. Suppose that the interference power at the receiver is I. If the receiver treats the interference similar to noise, then assuming Gaussian noise and interference, we get

$$C = W \log_2 (1 + SINR) \qquad \text{in bits/second} \tag{2.9}$$

where $SINR = \frac{P}{I+N_0W}$, is the signal-to-interference-and-noise ratio. As the capacity expression suggests, for fixed $I + N_0W$, capacity can be improved by increasing the transmit power, which will result in higher received power P, increasing the SINR. SINR in decibel notation is obtained as $10 \log \left(\frac{P}{I+N_0W}\right)$ dB.

Bandwidth, Capacity, Bit-Rate and Throughput: The terms *bandwidth*, *capacity*, *transmission rate*, and *throughput* are often used differently by different user/research communities. In this book, we will use these terms as follows. *Bandwidth*, as defined earlier, is a measure of the amount of spectrum occupied by a signal transmitted on a channel. Bandwidth is measured in units of Hertz (Hz). *Capacity* is the maximum possible rate of reliable information delivery. *Transmission rate* is the rate at which information is transmitted on the channel. We will sometimes use the term *bit-rate* to refer to transmission rate. *Throughput* is the rate at which data is reliably delivered between a pair of peers. Capacity, transmission rate and throughput can all be expressed in the units of bits-per-second (bps). As an example, a host S may transmit to host R using a IEEE 802.11g device, with transmission rate 54 Mbps. The bandwidth used in this case is 22 MHz, with the center frequency of approximately 2.4 GHz. The throughput achieved for transmissions from S to R may possibly be only 10 Mbps. Why would the throughput smaller than the transmission rate? There are overheads incurred during transmission, such as overhead of packet headers, and medium access control overhead (for instance, as per the protocol specification, host S cannot transmit all the time, and must remain idle intermittently). Also, packet losses may occur due to interference from other transmitters. For similar reasons, the *capacity* of the wireless channel would also be higher than the achieved throughput. The goal is to design the system such that the achieved throughput approaches the capacity.

2.6 Link Formation

Our discussion of binary PAM demodulation showed that the bit error probability depends on E_b/N_0. Although we ignored interference in our analysis, it should be clear that greater interference will generally result in higher error probability. What does this tell us about "link reliability" or the probability that a packet transmission on a link will be received reliably?

- Since error probability decreases with increasing E_b, higher transmission power will result in higher link reliability.
- Decreasing the transmission rate, while maintaining the transmit power constant, will result in higher energy-per-bit, improving link reliability.
- Increasing the packet size, while keeping all other parameters fixed (including the amount of redundancy for ECC), will decrease link reliability, since there are more bits in the packets that could be received erroneously.
- Poorer channel conditions (or, small a in Equation 2.6) will result in lower received power and lower E_b, which will reduce link reliability.

The channel conditions and interference can vary over time and space both, causing the link reliability to also vary over time and space. This has two consequences that affect design and performance of practical protocols.

- A host may reliably receive some packet transmissions from another host, but may not reliably receive other packet transmissions from the same host.

  When should we say that a wireless link "exists" between a pair of hosts? The abstraction of a "link" is often useful in designing higher layer protocols, particularly, routing protocols. As the above discussion suggests, there is no way to guarantee that a packet transmission between a pair of hosts will *always* be received reliably. Thus, we have to allow for unreliable transmissions on any such "link". For future reference, we will say that a link exists from host A to host B, if reliable packet transmissions can be performed from host A to host B with a sufficiently high probability. Similarly, a previously existing link from host A to host B would be said to be "broken" when its reliability becomes inadequate. Recall from the PAM discussion that this reliability depends on the packet size as well as the transmission rate, thus the "adequate reliability" is implicitly assumed to be for a certain set of operating parameters, such as transmission rate, power, and packet size.

  Conversely, when we say that a link does *not* exist from host A to host B, it does not necessarily mean that signals from host A do not propagate to host B causing interference at B, or that host B can never reliably receive A's transmission. Rather it simply means that reliability of transmission from A to B is not sufficiently high under the chosen operating parameters.

  When a link is deemed to be present between from host A to host B, we would say that B is host A's neighbor.

- One host at distance d from a transmitter may receive a packet transmission reliably, but another host within distance d may not receive the same transmission reliably. Due to this phenomenon, we cannot assume that all hosts within some constant distance of a host will receive its transmissions reliably. In other words, a broadcast performed by a host is not always received reliably by all the nearby hosts.

2.7 SINR-Threshold Model

The discussion Section 2.4 suggests that the probability of reliable reception of a packet is a function of the SINR at the receiver. In our discussion in later chapters, we will find it convenient to use an approximate model to determine whether a packet transmission is reliable or not. In particular, this approximate model, referred to as the *SINR-threshold*

model, makes the simplifying assumption that a packet will always be received reliably if the SINR at the receiver exceeds a certain threshold, say, β. In reality, even with SINR greater than β, the packet may be sometimes received unreliably. Thus, the SINR-threshold model should be viewed as a deterministic approximation of a non-deterministic phenomenon.

2.8 Path Gain and Path Loss

To calculate SINR, it is useful to know the received signal power. The received signal power is a function of the channel conditions, and the transmit power. We now introduce *path gain* to characterize the channel conditions. In particular, if transmit signal power is P_s and the received power for this signal is P_r, then *path gain* g is defined as

$$g = \frac{P_r}{P_s}$$

Clearly, path gain g is dependent on the channel gain. Inverse of path gain is called *path loss*. Thus, path loss is obtained as $\frac{P_s}{P_r}$. Despite the use of the word *gain* in the term *path gain*, the received power P_r is never greater than the transmit power P_s.

The path loss can be divided into two components, large scale path loss and small scale path loss. Large scale path loss can be viewed as a measure of the average channel conditions at a given location, with the small scale component characterizing the variation around the average.

In an attempt to describe the large scale behavior of the path loss, several different large scale path loss models have been introduced. We summarize a few such models here.

Free Space Propagation Model:

Free space propagation model determines received signal power in the "free space" environment wherein there is a line-of-sight (LoS) path between transmitter and receiver, and there are no other paths or obstacles between the two hosts. Let d denote the distance between the transmitter and the receiver. As per the free-space model,

$$P_r(d) \quad \propto \quad \frac{P_s}{d^2}$$

Considering that the surface area of a sphere at distance d from the transmitter is $4\pi d^2$, it should seem reasonable that the received power varies inversely with d^2. If we denote the constant of proportionality above as K, then

$$P_r(d) \quad = \quad K \, \frac{P_s}{d^2}$$

Note that the channel does not amplify the signal (that is, $P_r \leq P_s$). Thus, the path loss expression above cannot hold for "small" values of d. We will not formally define "small" distance d here, but we will assume that the above model holds for some suitably large d_0, and all $d \geq d_0$. Then it follows that, for $d \geq d_0$,

$$P_r(d) = P_r(d_0)\frac{d_0^2}{d^2}$$

In many applications (for instance, many wireless LANs), the distance between transmitter and receiver is indeed "large" enough, however, in certain applications (for instance, some RFID devices) the distance may be very small, and the above path loss model does not apply.

Recall that path loss is defined as $\frac{P_s}{P_r}$. Also, path loss in decibel (dB) is obtained as $10 \log \frac{P_s}{P_r}$. Let us denote path loss in dB at distance d as $PL(d)$. Then, for $d \geq d_0$, we have

$$PL(d) = PL(d_0) + 20 \log \frac{d}{d_0}$$

Two-Ray Ground Propagation Model:

Two-ray ground propagation model accounts for a direct line-of-sight (LoS) path, and a second path reflected from the ground. When the distance d between transmitter and receiver is large compared to antenna height from the ground, the two-ray model concludes that:

$$P_r \propto \frac{P_s}{d^4}$$

The exponent of d above is called the *path loss exponent*. The path loss exponent in this case is 4, larger than the path loss exponent of 2 for free space. Thus, signal attenuates much faster than in case of free space. For large enough d_0 and $d \geq d_0$,

$$P_r(d) = P_r(d_0)\frac{d_0^4}{d^4}$$

Log-Distance Path Loss Model:

Generalizing on the free space and two-ray models, we can obtain the following log-distance model, for a path loss exponent α. In this case, for large enough d_0 and $d \geq d_0$,

$$P_r(d) = P_r(d_0)\frac{d_0^\alpha}{d^\alpha}$$

and for path loss $PL(d)$ expressed in decibel,

$$PL(d) = PL(d_0) + 10\alpha \log \frac{d}{d_0}$$

Log-Normal Shadowing:

The above models only take into account the distance between the transmitter and the receiver. However, the path loss at a given distance from the transmitter is not constant, due to environmental obstructions. That is, all hosts at distance d from the transmitter do not see the same signal attenuation. Between the transmitter and a receiver A there may be a hill, whereas between the transmitter and another receiver B, the terrain may be flat. The path loss with "shadowing" caused by such obstructions is often modeled by the log-normal model. As per the log-normal model:

$$PL(d) = \overline{PL}(d) + X_\sigma$$

where $\overline{PL}(d)$ is the average large scale path loss at distance d, and X_σ is a Gaussian (normal) random variable with zero mean and standard deviation σ. In the above expression, X_σ, $PL(d)$ and $\overline{PL}(d)$ are in dB.

Many other path loss models have also been developed. For instance, some models account for additional path loss that occurs when signal travels through materials such as building walls (in contrast, the models listed above only account for the distance between the transmitter and the receiver).

Models versus Reality:

The various path loss models discussed above are approximations of reality. In general, it is unlikely that a simple model can capture the channel characteristics precisely. However, a simplified model can be potentially useful in analytical evaluation of performance of a system, and also in performing simulations, or to estimate system performance prior to deployment. When the exact channel characteristics may not be known (for instance, at design time), we are forced to rely on such models. As we will see later, for a protocol to be able to adapt to the channel characteristics, it is often useful to estimate the channel conditions dynamically, instead of relying on a simplified model of the channel.

2.8.1 Small-Scale Fading

The previous section discussed the models for large scale path loss. Large scale path loss characterizes the average path loss for the channel between a receiver and transmitter. *Small scale* effects introduce variations in the path loss. These variations may be over time and space both. The term "small" here refers to the scale of variations; the small scale variations cause relatively rapid changes in path loss over time and space, in contrast to the large scale effects. We briefly discuss two factors that lead to small scale variations.

- Movement of objects: Due to movement of objects (such as the transmitter, receiver or other obstacles in the environment) the length of the path traveled from the transmitter and the receiver changes with time, resulting in the Doppler effect.

- Multipath: In an environment containing many obstacles, such as an indoor environment or downtown of a city, there are many paths from the transmitter to the receiver that the signal may take, with the received signal being a composite of the signals received along the various paths. Different paths incur differing delays, thus the signals along the different paths do not necessarily arrive in phase.

Object movements and multipath cause *fading*, or relatively fast changes in the signal amplitude over time or space. Two fading models are in common use in wireless system evaluations:

- Rayleigh fading: This model is used when there are a large number of paths from the transmitter to the receiver, and none of them is dominant.
- Ricean fading: In contrast to Rayleigh fading, Ricean fading model is used to capture the presence of a dominant path.

For a given pair of hosts, say A and B, at a given instant of time, the path loss is identical in both directions, that is, regardless of whether A transmits to B, or B transmits to A (due to the principle of reciprocity), provided that the same antenna beamforms are used for transmission and reception. However, the path loss experienced by the transmissions from A and B (to each other) at different points of time can be different due to the large scale and small scale variations discussed above. Thus, transmissions from A to B and transmissions from B to A may potentially encounter different channel conditions, and different link reliability. In addition, differences in hardware implementation at the two hosts can result in different link characteristics in the two directions.

2.9 Summary

In this chapter, we defined the terms *bandwidth*, *capacity*, *path gain* and *path loss*. We also introduced a few path loss models. In addition, the chapter briefly discussed a simple modulation scheme, and the relationship between the bit error probability and the SNR. Understanding of these concepts related to the physical layer will help in understanding design decisions made in wireless protocol design.

APPENDIX A

Error Control Codes

As discussed in Section 2.2.4, error control codes are useful at different layers of the protocol stack, to perform error detection and error correction. For instance, as elaborated in Section 2.4, a receiver demodulates the received signal to obtain a sequence of bits as an estimate of the transmitted bit sequence. Let us refer to the demodulated bits as the *received bits*. As also seen in Section 2.4, due to the noise (or interference), the received bits may be erroneous. For instance, the transmitted bits may be 0111 whereas the received bits may be 0110. Unless additional precautions are taken, the receiver will not be able to recover the correct data. In fact, the receiver may not even detect that the received bits are in error. To allow a receiver to detect the errors, and possibly correct the errors, some *redundancy* must be incorporated in the transmitted bits. *Error control codes* (ECC) are used for this purpose.

For the purpose of illustration, Let us consider a simple error control code, which is an example of a family called the Hamming codes. The coding process for this example code is illustrated in Table A.1. In this case, the sender encodes 4 bits of data into a 7-bit codeword, and the code is said to be a (7,4) code. For instance, when the four data bits are 0010, the codeword is 0010 111. The *rate* of the code is $\frac{4}{7}$, since on average $\frac{4}{7}$ bits of data is encoded in each bit transmitted on the channel.

If the transmitter wants to send more than four bits while using the (7,4) Hamming code, then the transmitted bits are divided into block of four bits each, and each block encoded separately using the Hamming code. Similarly, at the receiver, the receiver will decode each received vector independently.

The 3 bits added during the encoding process are called *checkbits*. The redundant information encoded in the checkbits can be useful to correct errors. In fact, with this code, any single bit error can be corrected. The algorithm for error correction is quite simple: when a vector $\mathbf{v}$ is received (possibly containing an error), choose the codeword $\mathbf{u}$ from Table A.1 that differs from $\mathbf{v}$ in the fewest number of bits. The first four bits of $\mathbf{u}$ will be

the output of the decoder at the receiver. The number of bits in which $\mathbf{v}$ and $\mathbf{u}$ differ is said to be the hamming distance between $\mathbf{v}$ and $\mathbf{u}$.

The Hamming code has the interesting property that, if at most one transmitted bit becomes erroneous, then two conditions hold: (a) the transmitted codeword differs from the received vector in 1 bit, (b) all other codewords differ from the received vector in more than 1 bit. These conditions guarantee that, if at most one bit in the codeword is erroneous, the above *decoder* will choose the codeword $\mathbf{u}$, and the decoded data will be correct. For instance, suppose that the transmitted codeword is 0010 111 and the most significant bit is corrupted during transmission, such that the received vector is 1010 111. Observe that the transmitted codeword is, in fact, the closest codeword. In particular, the received vector differs only in one bit from the transmitted codeword, and at least in 2 bit positions from every other codeword. Thus, the single transmission error can be corrected by assuming that the codeword closest to the received vector is the correct codeword.

How does error correction help in improving reliability of transmission? Suppose that our "packet" contains 4 bits of data, and suppose that bit error probability is p. If we do not use any error control code, then the probability that the 4 bit data packet will be received *unreliably* is given by $1-(1-p)^4$. If $p = 10^{-3}$ then the packet unreliability is $1-(1-p)^4 = 0.004$. On the other hand, if we use the (7,3) single-error correcting (SEC) code, then the packet unreliability is given by $1-(1-p)^7 - 7p(1-p)^6 = 0.00002$. Clearly, the use of error correcting code can improve reliability of transmissions.

While the above Hamming code can correct single bit errors, it cannot correct two errors. For instance, assume that the transmitted codeword is 0010 111, and that the two most significant received bits are erroneous. Thus the received vector will be 1110 111, which will be decoded to the closest codeword 1111 111, with the decoded data being 1111. Clearly, in this case, the decoded data is erroneous. The receiver will use decoded data 1111 without realizing that the data is incorrect. In general, which error patterns can be corrected depends on the set of codewords used. More redundancy generally allows the decoder to correct more errors.

It is also possible to design codes that allow the receiver to correct some of the errors, and detect a larger class of errors. For instance, Table A.2 shows an (8,4) code obtained by adding an even parity bit to the codewords in the (7,4) code in Table A.1. In this code, any pair of codewords differs in at least 4 bit positions. This property can be exploited to design a decoding algorithm that can obtain the correct codeword if at most 1 received bit is erroneous, and detect the presence of 2 erroneous bits. Thus, the code is said to be a *single error-correcting double error-detecting* (SEC-DED) code. In particular, here is the decoding algorithm to be applied to received vector $\mathbf{v}$:

- If a codeword $\mathbf{u}$ differs from $\mathbf{v}$ in at most 1 bit position, assume that $\mathbf{u}$ is the transmitted codeword.

Data	Codeword
0000	0000 000
0001	0001 011
0010	0010 111
0011	0011 100
0100	0100 110
0101	0101 101
0110	0110 001
0111	0111 010
1000	1000 101
1001	1001 110
1010	1010 010
1011	1011 001
1100	1100 011
1101	1101 000
1110	1110 100
1111	1111 111

Figure A.1 (7,4) Hamming code

- If all codewords differ from **v** in at least two bit positions, then declare that more than 1 received bit is erroneous. In this case, the receiver will reject the received vector, and not produce any decoded data.

The (8,4) code includes greater redundancy than the (7,4) code, and the (8,4) code improves on the (7,4) code by being able to detect two errors. But the new code still cannot correct more than one error. Also, the above algorithm can result in incorrect outcome if more than two errors exist in the received vector.

For a given code, we can trade-off its ability to correct errors with its ability to detect errors. In particular, again consider the (8,4) code. We used it above as a SEC-DED code. Instead, we can also use the code to perform *only* error detection. In this case, we will not attempt to perform any error correction, but as a trade-off, we will now be able to detect up to 3 errors. The decoding algorithm for this purpose is simple: (a) if the received vector is identical to a codeword, decode to this codeword, (b) else declare that the received vector is erroneous.

The above simple examples make three important points:

- The ability to detect and correct errors is a function of the redundancy incorporated in the code. A greater capability requires greater levels of redundancy.

Data	Codeword	
0000	0000	0000
0001	0001	0111
0010	0010	1110
0011	0011	1001
0100	0100	1101
0101	0101	1010
0110	0110	0011
0111	0111	0100
1000	1000	1011
1001	1001	1100
1010	1010	0101
1011	1011	0010
1100	1100	0110
1101	1101	0001
1110	1110	1001
1111	1111	1111

Figure A.2 (8,4) Single error-correcting, double error-detecting code

- The ability to detect errors and correct errors can be traded with each other. With a given level of redundancy, in order to be able to improve the probability of error detection, we have to be willing to accept a lower probability of obtaining correct decoded data.

- No error control code can correct an arbitrary set of errors. Similarly, no useful error control code can detect an arbitrary set of errors.

The last observation above has led to the use of error control codes (ECC) at various levels of the protocol stack. In particular, while the higher layers of the protocol stack benefit from the error control capabilities incorporated at the lowers layers, the higher layers usually include some error detection and/or correction capabilities as well, to protect against the possibility that the lower layer error control may fail. We discuss some examples:

- The IEEE 802.11a physical layer uses a *convolutional code* to perform error correction. As noted earlier, some errors patterns in the received bits may result in incorrect decoding. Thus, the higher layers need to use additional error protection.

- The IEEE 802.11 MAC header includes *cyclic redundancy check*, which can be used to detect some errors in a MAC layer frame.

- Despite the use of error correction and detection mechanisms at the physical and link layers, the packets received by the network layer (IP) may still contain errors. In addition, errors may also be inserted into a packet due to hardware failure (for instance, due to memory corruption in a buffer at a router). To facilitate detection of such errors, IP uses a *header checksum* in each IP header. This checksum only allows detection of errors in the IP header, offering no protection to the data in an IP datagram. If errors are detected in the IP header, the datagram is discarded – this event is perceived as a *packet loss* by the higher layers.

- TCP header also contains a checksum that covers the TCP header as well as the data in a TCP segment. This checksum is used only for error detection. If errors are detected in the TCP segment, the segment is discarded, and this event is viewed as a *packet loss* by the TCP receiver.

- The ultimate responsibility for achieving reliability rests at the application layer. Therefore, many applications incorporate additional application-layer mechanisms to achieve the desired level of reliability. For instance, consider an application that needs to transmit data reliably from source computer A to computer B. This goal may possibly be achieved by using a TCP connection between A and B, by relying on TCP's retransmission mechanism. An alternative is to use a UDP connection from A to B. In this case, however, some of the packets sent by A may be lost, either due to detection of errors at the lower layers (for instance, using the IP checksum), or due to buffer

overflow at an intermediate host. To allow recovery of the data despite such packet losses, the sender host A can send the data packets, followed by redundant packets obtained as a function of the data packets. If the redundant packets are designed correctly, then after a sufficient number of redundant packet transmissions, the recipient can recover all the data packets, despite the loss of a small number of such packets.

Despite the use of error control codes at the various layers of the protocol stack, there is always a non-zero probability that some errors may escape detection by all the error control mechanisms. In practice, this probability can be made small enough by using adequate redundancy at the different layers of the protocol stack.

APPENDIX B

Frequency-Domain Representation of Signals

The frequency-domain representation discussed here associates complex "weights" corresponding to each frequency component in the signal. Therefore, we begin this section with a brief discussion of complex numbers. A complex number has two parts: *real* and *imaginary*. For instance, in the complex number $a + jb$, a is the real part and b is the imaginary part. j represents $\sqrt{-1}$ (the imaginary unit). Complex number $a + jb$ can be represented on a two-dimensional graph as depicted in Figure B.1(a), where the horizontal axis corresponds to the real part of the complex number, and the vertical axis corresponds to the imaginary part. An alternative representation for $a + jb$ is using a vector pointing from the origin to $a + jb$, as shown in Figure B.1(b). Let ϕ denote the angle made by this vector with the positive direction of the horizontal axis, and let m denote the length of the vector. m and ϕ are the *magnitude* and *phase* of the vector, respectively. It follows that

$$\begin{aligned} m &= \sqrt{a^2 + b^2} \\ a &= m\cos\phi \\ b &= m\sin\phi \\ a + jb &= m(\cos\phi + j\sin\phi) \end{aligned}$$

Euler's formula tells us that $\cos\phi + j\sin\phi = e^{j\phi}$. Thus,

$$a + jb = me^{j\phi}$$

Now, what if ϕ is a function of time? Specifically, let us consider

$$me^{j(2\pi ft+\theta)}$$

where t represents time, and m represents magnitude (thus, t and m are both non-negative). f and θ are constants, whose significance should become clearer soon. Recall that

$$me^{j(2\pi ft+\theta)} = m\cos(2\pi ft + \theta) + jm\sin(2\pi ft + \theta)$$

The vector representation for $me^{j(2\pi ft+\theta)}$ will consist of a vector of length m making an angle of $2\pi ft + \theta$ with the horizontal axis. The vector at time $t = 0$ is shown in Figure B.1(c), assuming that $\theta = -\pi/4$. Figure B.1(d) depicts the vector at time $t = \frac{1}{4f}$, when f is positive, and Figure B.1(e) depicts the vector at time $t = \frac{-1}{4f}$, when f is negative.

You should try this for other values of time t. It should be clear that the vector rotates counter-clockwise with increasing time when f is positive, and clockwise when f is negative. The vector rotates one full circle for each time interval of duration $1/|f|$, thus rotating $|f|$ times per second. We will refer to f as the frequency – thus, the frequency can also be negative.

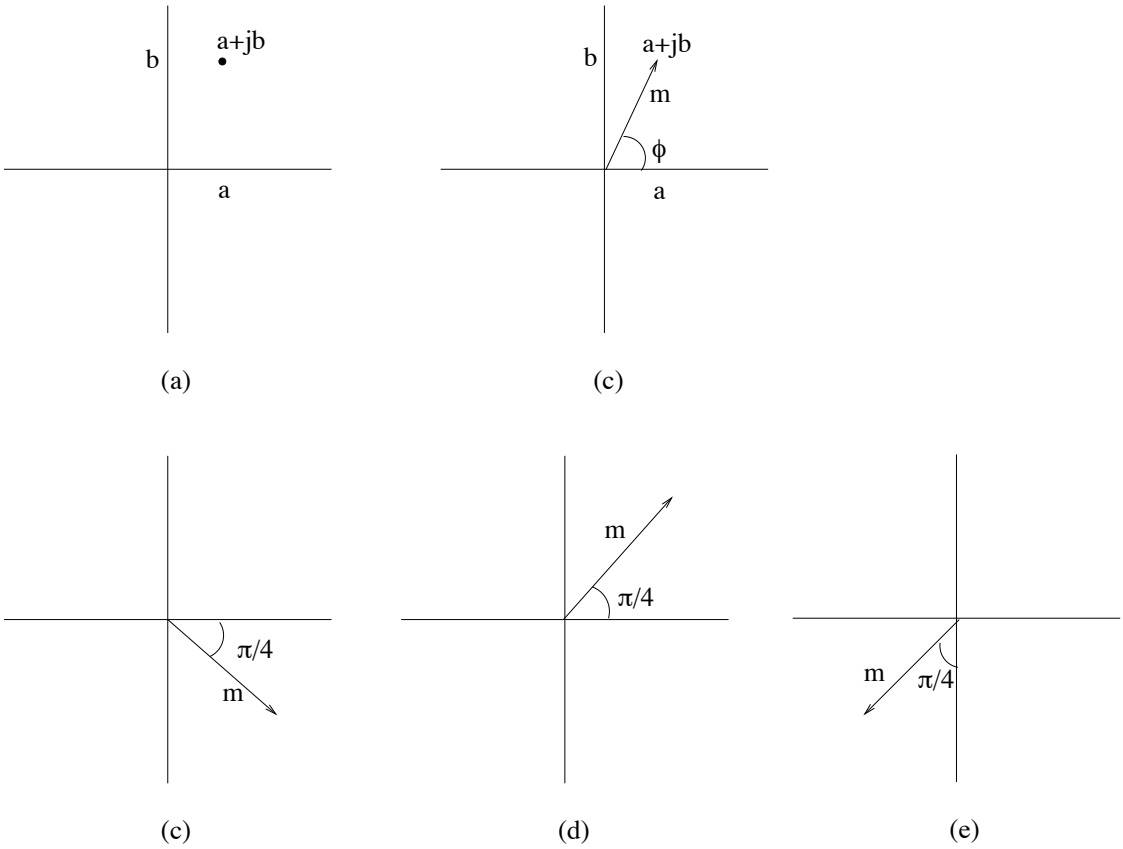

Figure B.1 Vector representation of complex numbers

Euler's formula implies that $j = e^{j\frac{\pi}{2}}$, $-j = e^{j\frac{3\pi}{2}}$, and

$$\sin(2\pi ft) = \frac{j}{2}e^{-j2\pi ft} - \frac{j}{2}e^{j2\pi ft}$$

$\cos(2\pi ft)$ can be represented similar to the sin function above. Specifically,

$$\cos(2\pi ft) = \frac{1}{2}e^{-j2\pi ft} + \frac{1}{2}e^{j2\pi ft}$$

Then it follows that any signal that can be represented as a linear combination of sin and cos functions of time can also be represented as a linear combination of terms of the form $e^{j2\pi ft}$. For instance, function

$$h(t) = \sin(2\pi f_0 t + \phi_0) + \cos(2\pi f_1 t + \phi_1)$$

can be represented as

$$\begin{aligned} h(t) &= \left[\frac{j}{2}e^{j(-2\pi f_0 t - \phi_0)} - \frac{j}{2}e^{j(2\pi f_0 t + \phi_0)}\right] + \left[\frac{1}{2}e^{j(-2\pi f_1 t - \phi_1)} + \frac{1}{2}e^{j(2\pi f_1 t + \phi_1)}\right] \quad \text{(B.1)} \\ &= H(-f_0)\ e^{j2\pi(-f_0)t} + H(f_0)\ e^{j2\pi f_0 t} + H(-f_1)\ e^{j2\pi(-f_1)t} + H(f_1)\ e^{j2\pi f_1 t} \quad \text{(B.2)} \end{aligned}$$

where $H(-f_0) = \frac{j}{2}e^{-j\phi_0}$, $H(f_0) = -\frac{j}{2}e^{j\phi_0}$, $H(-f_1) = \frac{1}{2}e^{-j\phi_1}$, and $H(f_1) = \frac{1}{2}e^{j\phi_1}$. This is said to be a frequency-domain representation of the signal. Observe that the $H(f)$ *coefficients* above are complex numbers (in other cases, the coefficients may also be real numbers). Note that frequency f in $H(f)$ may be negative or non-negative. The coefficients $H(f)$ above are "weights" that determine the contribution of frequency f to the function $h(t)$.

While the above approach of representing a function of time as a discrete sum of terms of the form $H(f)\ e^{j2\pi ft}$ suffices for some signals, the methodology can be generalized to obtain frequency-domain representation for a larger class of signals. This approach, namely Fourier Transform, is quite useful in digital signal processing. Fourier transform of a signal is a way of representing the manner in which the signal "occupies" the frequency spectrum. Specifically, if $X(f)$ is the Fourier transform of a signal $x(t)$, then $x(t)$ can be obtained using the following inverse Fourier transform operation:

$$x(t) = \int_{-\infty}^{+\infty} X(f)e^{j2\pi ft}\, df \quad \text{(B.3)}$$

f in $X(f)$ denotes frequency, and t in $x(t)$ denotes time. Thus, $X(f)$ is the frequency-domain representation of a signal, and $x(t)$ is the time-domain representation of the same signal. In general, Fourier transform $X(f)$ may be a complex number. f may be positive or negative. Notice that the simple addition of the terms of the form $H(f)\ e^{j2\pi ft}$ in Equation B.2 has been replaced by an integral in Equation B.3.