FOURTH EDITION

Computer Networks

Virtual LAN

ANDREW S. TANENBAUM

Computer Networks

Fourth Edition

Andrew S. Tanenbaum

Vrije Universiteit
Amsterdam, The Netherlands

PH
PTR

Prentice Hall PTR
Upper Saddle River, NJ 07458
www.phptr.com

Library of Congress Cataloging-in-Publication Data

Tanenbaum, Andrew S.
Computer networks / Andrew S. Tanenbaum.--4th ed.
p. cm.
Includes bibliographical references.
ISBN 0-13-066102-3
1. Computer networks. I. Title.
TK5105.5 .T36 2002
004.6--dc21 2002029263

Editorial/production supervision: *Patti Guerrieri*
Cover design director: *Jerry Votta*
Cover designer: *Anthony Gemmellaro*
Cover design: *Andrew S. Tanenbaum*
Art director: *Gail Cocker-Bogusz*
Interior Design: *Andrew S. Tanenbaum*
Interior graphics: *Hadel Studio*
Typesetting: *Andrew S. Tanenbaum*
Manufacturing buyer: *Maura Zaldivar*
Executive editor: *Mary Franz*
Editorial assistant: *Noreen Regina*
Marketing manager: *Dan DePasquale*

PH PTR

Publishing as Prentice Hall PTR
Upper Saddle River, New Jersey 07458

Prentice Hall books are widely used by corporations and government agencies for training, marketing, and resale.

For information regarding corporate and government bulk discounts please contact:
Corporate and Government Sales (800) 382-3419 or corpsales@pearsontechgroup.com

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-066102-3

Pearson Education LTD.
Pearson Education Australia PTY, Limited
Pearson Education Singapore, Pte. Ltd.
Pearson Education North Asia Ltd.
Pearson Education Canada, Ltd.
Pearson Educación de Mexico, S.A. de C.V.
Pearson Education — Japan
Pearson Education Malaysia, Pte. Ltd.

2.6 THE MOBILE TELEPHONE SYSTEM

The traditional telephone system (even if it some day gets multigigabit end-to-end fiber) will still not be able to satisfy a growing group of users: people on the go. People now expect to make phone calls from airplanes, cars, swimming pools, and while jogging in the park. Within a few years they will also expect to send e-mail and surf the Web from all these locations and more. Consequently, there is a tremendous amount of interest in wireless telephony. In the following sections we will study this topic in some detail.

Wireless telephones come in two basic varieties: cordless phones and mobile phones (sometimes called **cell phones**). **Cordless phones** are devices consisting of a base station and a handset sold as a set for use within the home. These are never used for networking, so we will not examine them further. Instead we will concentrate on the mobile system, which is used for wide area voice and data communication.

Mobile phones have gone through three distinct generations, with different technologies:

1. Analog voice.
2. Digital voice.
3. Digital voice and data (Internet, e-mail, etc.).

Although most of our discussion will be about the technology of these systems, it is interesting to note how political and tiny marketing decisions can have a huge impact. The first mobile system was devised in the U.S. by AT&T and mandated for the whole country by the FCC. As a result, the entire U.S. had a single (analog) system and a mobile phone purchased in California also worked in New York. In contrast, when mobile came to Europe, every country devised its own system, which resulted in a fiasco.

Europe learned from its mistake and when digital came around, the government-run PTTs got together and standardized on a single system (GSM), so any European mobile phone will work anywhere in Europe. By then, the U.S. had decided that government should not be in the standardization business, so it left digital to the marketplace. This decision resulted in different equipment manufacturers producing different kinds of mobile phones. As a consequence, the U.S. now has two major incompatible digital mobile phone systems in operation (plus one minor one).

Despite an initial lead by the U.S., mobile phone ownership and usage in Europe is now far greater than in the U.S. Having a single system for all of Europe is part of the reason, but there is more. A second area where the U.S. and Europe differed is in the humble matter of phone numbers. In the U.S. mobile phones are mixed in with regular (fixed) telephones. Thus, there is no way for a caller to see if, say, (212) 234-5678 is a fixed telephone (cheap or free call) or a mobile phone (expensive call). To keep people from getting nervous about using the telephone, the telephone companies decided to make the mobile phone owner pay for incoming calls. As a consequence, many people hesitated to buy a mobile phone for fear of running up a big bill by just receiving calls. In Europe, mobile phones have a special area code (analogous to 800 and 900 numbers) so they are instantly recognizable. Consequently, the usual rule of "caller pays" also applies to mobile phones in Europe (except for international calls where costs are split).

A third issue that has had a large impact on adoption is the widespread use of prepaid mobile phones in Europe (up to 75% in some areas). These can be purchased in many stores with no more formality than buying a radio. You pay and you go. They are preloaded with, for example, 20 or 50 euro and can be recharged (using a secret PIN code) when the balance drops to zero. As a consequence, practically every teenager and many small children in Europe have (usually prepaid) mobile phones so their parents can locate them, without the danger of the child running up a huge bill. If the mobile phone is used only occasionally, its use is essentially free since there is no monthly charge or charge for incoming calls.

2.6.1 First-Generation Mobile Phones: Analog Voice

Enough about the politics and marketing aspects of mobile phones. Now let us look at the technology, starting with the earliest system. Mobile radiotelephones were used sporadically for maritime and military communication during the early decades of the 20th century. In 1946, the first system for car-based telephones was set up in St. Louis. This system used a single large transmitter on top of a tall building and had a single channel, used for both sending and receiving. To talk, the user had to push a button that enabled the transmitter and disabled the receiver. Such systems, known as **push-to-talk systems**, were installed in several cities beginning in the late 1950s. CB-radio, taxis, and police cars on television programs often use this technology.

In the 1960s, **IMTS** (**Improved Mobile Telephone System**) was installed. It, too, used a high-powered (200-watt) transmitter, on top of a hill, but now had two frequencies, one for sending and one for receiving, so the push-to-talk button was no longer needed. Since all communication from the mobile telephones went inbound on a different channel than the outbound signals, the mobile users could not hear each other (unlike the push-to-talk system used in taxis).

IMTS supported 23 channels spread out from 150 MHz to 450 MHz. Due to the small number of channels, users often had to wait a long time before getting a dial tone. Also, due to the large power of the hilltop transmitter, adjacent systems had to be several hundred kilometers apart to avoid interference. All in all, the limited capacity made the system impractical.

Advanced Mobile Phone System

All that changed with **AMPS (Advanced Mobile Phone System)**, invented by Bell Labs and first installed in the United States in 1982. It was also used in England, where it was called TACS, and in Japan, where it was called MCS-L1. Although no longer state of the art, we will look at it in some detail because many of its fundamental properties have been directly inherited by its digital successor, D-AMPS, in order to achieve backward compatibility.

In all mobile phone systems, a geographic region is divided up into **cells**, which is why the devices are sometimes called cell phones. In AMPS, the cells are typically 10 to 20 km across; in digital systems, the cells are smaller. Each cell uses some set of frequencies not used by any of its neighbors. The key idea that gives cellular systems far more capacity than previous systems is the use of relatively small cells and the reuse of transmission frequencies in nearby (but not adjacent) cells. Whereas an IMTS system 100 km across can have one call on each frequency, an AMPS system might have 100 10-km cells in the same area and be able to have 10 to 15 calls on each frequency, in widely separated cells. Thus, the cellular design increases the system capacity by at least an order of magnitude, more as the cells get smaller. Furthermore, smaller cells mean that less power is needed, which leads to smaller and cheaper transmitters and handsets. Hand-held telephones put out 0.6 watts; transmitters in cars are 3 watts, the maximum allowed by the FCC.

The idea of frequency reuse is illustrated in Fig. 2-41(a). The cells are normally roughly circular, but they are easier to model as hexagons. In Fig. 2-41(a), the cells are all the same size. They are grouped in units of seven cells. Each letter indicates a group of frequencies. Notice that for each frequency set, there is a buffer about two cells wide where that frequency is not reused, providing for good separation and low interference.

Finding locations high in the air to place base station antennas is a major issue. This problem has led some telecommunication carriers to forge alliances with the Roman Catholic Church, since the latter owns a substantial number of exalted potential antenna sites worldwide, all conveniently under a single management.

In an area where the number of users has grown to the point that the system is overloaded, the power is reduced, and the overloaded cells are split into smaller **microcells** to permit more frequency reuse, as shown in Fig. 2-41(b). Telephone companies sometimes create temporary microcells, using portable towers with satellite links at sporting events, rock concerts, and other places where large numbers of mobile users congregate for a few hours. How big the cells should be is a complex matter, which is treated in (Hac, 1995).

At the center of each cell is a base station to which all the telephones in the cell transmit. The base station consists of a computer and transmitter/receiver connected to an antenna. In a small system, all the base stations are connected to

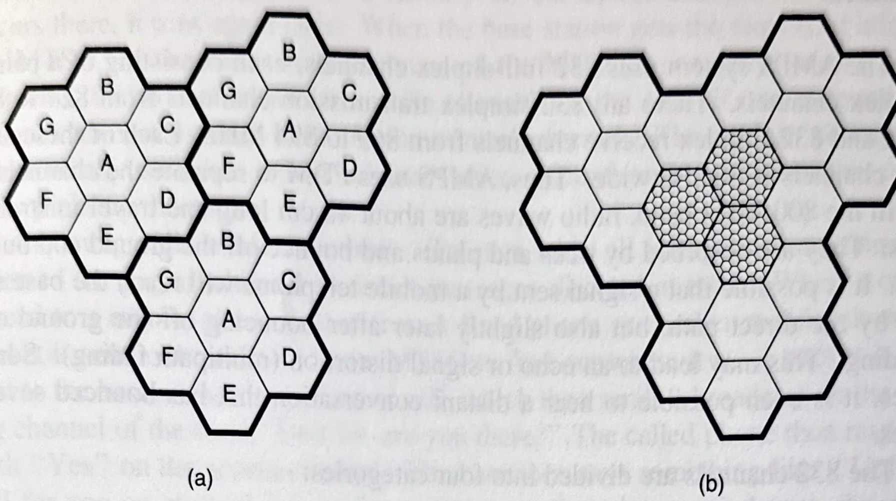

Figure 2-41. (a) Frequencies are not reused in adjacent cells. (b) To add more users, smaller cells can be used.

a single device called an **MTSO (Mobile Telephone Switching Office)** or **MSC (Mobile Switching Center)**. In a larger one, several MTSOs may be needed, all of which are connected to a second-level MTSO, and so on. The MTSOs are essentially end offices as in the telephone system, and are, in fact, connected to at least one telephone system end office. The MTSOs communicate with the base stations, each other, and the PSTN using a packet-switching network.

At any instant, each mobile telephone is logically in one specific cell and under the control of that cell's base station. When a mobile telephone physically leaves a cell, its base station notices the telephone's signal fading away and asks all the surrounding base stations how much power they are getting from it. The base station then transfers ownership to the cell getting the strongest signal, that is, the cell where the telephone is now located. The telephone is then informed of its new boss, and if a call is in progress, it will be asked to switch to a new channel (because the old one is not reused in any of the adjacent cells). This process, called **handoff**, takes about 300 msec. Channel assignment is done by the MTSO, the nerve center of the system. The base stations are really just radio relays.

Handoffs can be done in two ways. In a **soft handoff**, the telephone is acquired by the new base station before the previous one signs off. In this way there is no loss of continuity. The downside here is that the telephone needs to be able to tune to two frequencies at the same time (the old one and the new one). Neither first nor second generation devices can do this.

In a **hard handoff**, the old base station drops the telephone before the new one acquires it. If the new one is unable to acquire it (e.g., because there is no available frequency), the call is disconnected abruptly. Users tend to notice this, but it is inevitable occasionally with the current design.

Channels

The AMPS system uses 832 full-duplex channels, each consisting of a pair of simplex channels. There are 832 simplex transmission channels from 824 to 849 MHz and 832 simplex receive channels from 869 to 894 MHz. Each of these simplex channels is 30 kHz wide. Thus, AMPS uses FDM to separate the channels.

In the 800-MHz band, radio waves are about 40 cm long and travel in straight lines. They are absorbed by trees and plants and bounce off the ground and buildings. It is possible that a signal sent by a mobile telephone will reach the base station by the direct path, but also slightly later after bouncing off the ground or a building. This may lead to an echo or signal distortion (multipath fading). Sometimes, it is even possible to hear a distant conversation that has bounced several times.

The 832 channels are divided into four categories:

1. Control (base to mobile) to manage the system.
2. Paging (base to mobile) to alert mobile users to calls for them.
3. Access (bidirectional) for call setup and channel assignment.
4. Data (bidirectional) for voice, fax, or data.

Twenty-one of the channels are reserved for control, and these are wired into a PROM in each telephone. Since the same frequencies cannot be reused in nearby cells, the actual number of voice channels available per cell is much smaller than 832, typically about 45.

Call Management

Each mobile telephone in AMPS has a 32-bit serial number and a 10-digit telephone number in its PROM. The telephone number is represented as a 3-digit area code in 10 bits, and a 7-digit subscriber number in 24 bits. When a phone is switched on, it scans a preprogrammed list of 21 control channels to find the most powerful signal.

The phone then broadcasts its 32-bit serial number and 34-bit telephone number. Like all the control information in AMPS, this packet is sent in digital form, multiple times, and with an error-correcting code, even though the voice channels themselves are analog.

When the base station hears the announcement, it tells the MTSO, which records the existence of its new customer and also informs the customer's home MTSO of his current location. During normal operation, the mobile telephone reregisters about once every 15 minutes.

To make a call, a mobile user switches on the phone, enters the number to be called on the keypad, and hits the SEND button. The phone then transmits the number to be called and its own identity on the access channel. If a collision occurs there, it tries again later. When the base station gets the request, it informs the MTSO. If the caller is a customer of the MTSO's company (or one of its partners), the MTSO looks for an idle channel for the call. If one is found, the channel number is sent back on the control channel. The mobile phone then automatically switches to the selected voice channel and waits until the called party picks up the phone.

Incoming calls work differently. To start with, all idle phones continuously listen to the paging channel to detect messages directed at them. When a call is placed to a mobile phone (either from a fixed phone or another mobile phone), a packet is sent to the callee's home MTSO to find out where it is. A packet is then sent to the base station in its current cell, which then sends a broadcast on the paging channel of the form "Unit 14, are you there?" The called phone then responds with "Yes" on the access channel. The base then says something like: "Unit 14, call for you on channel 3." At this point, the called phone switches to channel 3 and starts making ringing sounds (or playing some melody the owner was given as a birthday present).

2.6.2 Second-Generation Mobile Phones: Digital Voice

The first generation of mobile phones was analog; the second generation was digital. Just as there was no worldwide standardization during the first generation, there was also no standardization during the second, either. Four systems are in use now: D-AMPS, GSM, CDMA, and PDC. Below we will discuss the first three. PDC is used only in Japan and is basically D-AMPS modified for backward compatibility with the first-generation Japanese analog system. The name **PCS** (**Personal Communications Services**) is sometimes used in the marketing literature to indicate a second-generation (i.e., digital) system. Originally it meant a mobile phone using the 1900 MHz band, but that distinction is rarely made now.

D-AMPS—The Digital Advanced Mobile Phone System

The second generation of the AMPS systems is **D-AMPS** and is fully digital. It is described in International Standard IS-54 and its successor IS-136. D-AMPS was carefully designed to co-exist with AMPS so that both first- and second-generation mobile phones could operate simultaneously in the same cell. In particular, D-AMPS uses the same 30 kHz channels as AMPS and at the same frequencies so that one channel can be analog and the adjacent ones can be digital. Depending on the mix of phones in a cell, the cell's MTSO determines which channels are analog and which are digital, and it can change channel types dynamically as the mix of phones in a cell changes.

When D-AMPS was introduced as a service, a new frequency band was made available to handle the expected increased load. The upstream channels were in the 1850–1910 MHz range, and the corresponding downstream channels were in the 1930–1990 MHz range, again in pairs, as in AMPS. In this band, the waves are 16 cm long, so a standard ¼-wave antenna is only 4 cm long, leading to smaller phones. However, many D-AMPS phones can use both the 850-MHz and 1900-MHz bands to get a wider range of available channels.

On a D-AMPS mobile phone, the voice signal picked up by the microphone is digitized and compressed using a model that is more sophisticated than the delta modulation and predictive encoding schemes we studied earlier. Compression takes into account detailed properties of the human vocal system to get the bandwidth from the standard 56-kbps PCM encoding to 8 kbps or less. The compression is done by a circuit called a **vocoder** (Bellamy, 2000). The compression is done in the telephone, rather than in the base station or end office, to reduce the number of bits sent over the air link. With fixed telephony, there is no benefit to having compression done in the telephone, since reducing the traffic over the local loop does not increase system capacity at all.

With mobile telephony there is a huge gain from doing digitization and compression in the handset, so much so that in D-AMPS, three users can share a single frequency pair using time division multiplexing. Each frequency pair supports 25 frames/sec of 40 msec each. Each frame is divided into six time slots of 6.67 msec each, as illustrated in Fig. 2-42(a) for the lowest frequency pair.

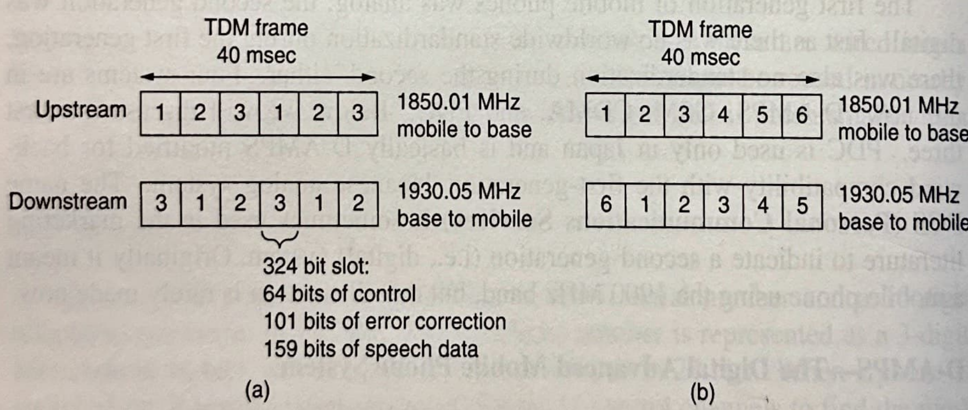

Figure 2-42. (a) A D-AMPS channel with three users. (b) A D-AMPS channel with six users.

Each frame holds three users who take turns using the upstream and downstream links. During slot 1 of Fig. 2-42(a), for example, user 1 may transmit to the base station and user 3 is receiving from the base station. Each slot is 324 bits long, of which 64 bits are used for guard times, synchronization, and control purposes, leaving 260 bits for the user payload. Of the payload bits, 101 are used for error correction over the noisy air link, so ultimately only 159 bits are left for compressed speech. With 50 slots/sec, the bandwidth available for compressed speech is just under 8 kbps, 1/7 of the standard PCM bandwidth.

Using better compression algorithms, it is possible to get the speech down to 4 kbps, in which case six users can be stuffed into a frame, as illustrated in Fig. 2-42(b). From the operator's perspective, being able to squeeze three to six times as many D-AMPS users into the same spectrum as one AMPS user is a huge win and explains much of the popularity of PCS. Of course, the quality of speech at 4 kbps is not comparable to what can be achieved at 56 kbps, but few PCS operators advertise their hi-fi sound quality. It should also be clear that for data, an 8 kbps channel is not even as good as an ancient 9600-bps modem.

The control structure of D-AMPS is fairly complicated. Briefly summarized, groups of 16 frames form a superframe, with certain control information present in each superframe a limited number of times. Six main control channels are used: system configuration, real-time and nonreal-time control, paging, access response, and short messages. But conceptually, it works like AMPS. When a mobile is switched on, it makes contact with the base station to announce itself and then listens on a control channel for incoming calls. Having picked up a new mobile, the MTSO informs the user's home base where he is, so calls can be routed correctly.

One difference between AMPS and D-AMPS is how handoff is handled. In AMPS, the MTSO manages it completely without help from the mobile devices. As can be seen from Fig. 2-42, in D-AMPS, 1/3 of the time a mobile is neither sending nor receiving. It uses these idle slots to measure the line quality. When it discovers that the signal is waning, it complains to the MTSO, which can then break the connection, at which time the mobile can try to tune to a stronger signal from another base station. As in AMPS, it still takes about 300 msec to do the handoff. This technique is called **MAHO** (**Mobile Assisted HandOff**).

GSM—The Global System for Mobile Communications

D-AMPS is widely used in the U.S. and (in modified form) in Japan. Virtually everywhere else in the world, a system called **GSM** (**Global System for Mobile communications**) is used, and it is even starting to be used in the U.S. on a limited scale. To a first approximation, GSM is similar to D-AMPS. Both are cellular systems. In both systems, frequency division multiplexing is used, with each mobile transmitting on one frequency and receiving on a higher frequency (80 MHz higher for D-AMPS, 55 MHz higher for GSM). Also in both systems, a single frequency pair is split by time-division multiplexing into time slots shared by multiple mobiles. However, the GSM channels are much wider than the AMPS channels (200 kHz versus 30 kHz) and hold relatively few additional users (8 versus 3), giving GSM a much higher data rate per user than D-AMPS.

Below we will briefly discuss some of the main properties of GSM. However, the printed GSM standard is over 5000 [sic] pages long. A large fraction of this material relates to engineering aspects of the system, especially the design of receivers to handle multipath signal propagation, and synchronizing transmitters and receivers. None of this will be even mentioned below.

Each frequency band is 200 kHz wide, as shown in Fig. 2-43. A GSM system has 124 pairs of simplex channels. Each simplex channel is 200 kHz wide and supports eight separate connections on it, using time division multiplexing. Each currently active station is assigned one time slot on one channel pair. Theoretically, 992 channels can be supported in each cell, but many of them are not available, to avoid frequency conflicts with neighboring cells. In Fig. 2-43, the eight shaded time slots all belong to the same connection, four of them in each direction. Transmitting and receiving does not happen in the same time slot because the GSM radios cannot transmit and receive at the same time and it takes time to switch from one to the other. If the mobile station assigned to 890.4/935.4 MHz and time slot 2 wanted to transmit to the base station, it would use the lower four shaded slots (and the ones following them in time), putting some data in each slot until all the data had been sent.

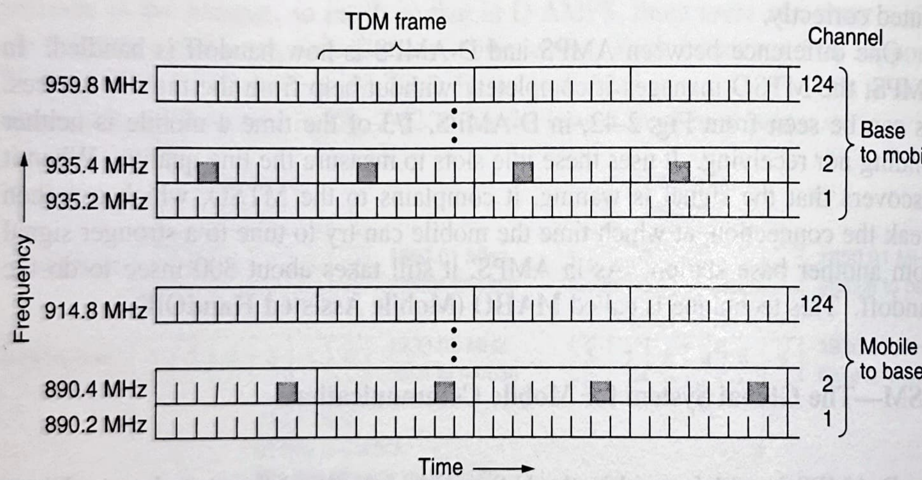

Figure 2-43. GSM uses 124 frequency channels, each of which uses an eight-slot TDM system.

The TDM slots shown in Fig. 2-43 are part of a complex framing hierarchy. Each TDM slot has a specific structure, and groups of TDM slots form multiframes, also with a specific structure. A simplified version of this hierarchy is shown in Fig. 2-44. Here we can see that each TDM slot consists of a 148-bit data frame that occupies the channel for 577 μsec (including a 30-μsec guard time after each slot). Each data frame starts and ends with three 0 bits, for frame delineation purposes. It also contains two 57-bit *Information* fields, each one having a control bit that indicates whether the following *Information* field is for voice or data. Between the *Information* fields is a 26-bit *Sync* (training) field that is used by the receiver to synchronize to the sender's frame boundaries.

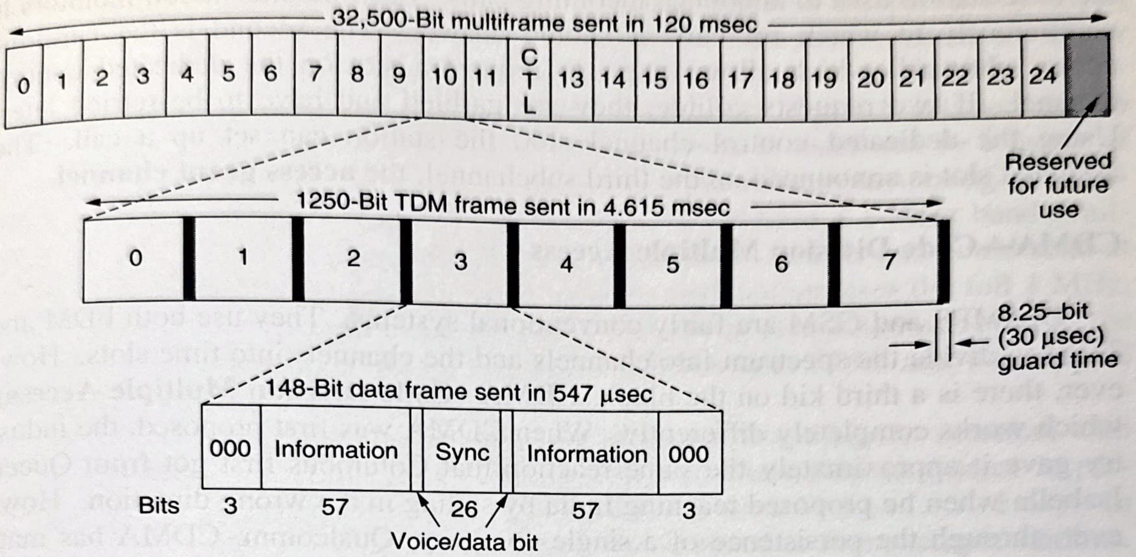

Figure 2-44. A portion of the GSM framing structure.

A data frame is transmitted in 547 μsec, but a transmitter is only allowed to send one data frame every 4.615 msec, since it is sharing the channel with seven other stations. The gross rate of each channel is 270,833 bps, divided among eight users. This gives 33.854 kbps gross, more than double D-AMPS' 324 bits 50 times per second for 16.2 kbps. However, as with AMPS, the overhead eats up a large fraction of the bandwidth, ultimately leaving 24.7 kbps worth of payload per user before error correction. After error correction, 13 kbps is left for speech, giving substantially better voice quality than D-AMPS (at the cost of using correspondingly more bandwidth).

As can be seen from Fig. 2-44, eight data frames make up a TDM frame and 26 TDM frames make up a 120-msec multiframe. Of the 26 TDM frames in a multiframe, slot 12 is used for control and slot 25 is reserved for future use, so only 24 are available for user traffic.

However, in addition to the 26-slot multiframe shown in Fig. 2-44, a 51-slot multiframe (not shown) is also used. Some of these slots are used to hold several control channels used to manage the system. The **broadcast control channel** is a continuous stream of output from the base station containing the base station's identity and the channel status. All mobile stations monitor their signal strength to see when they have moved into a new cell.

The **dedicated control channel** is used for location updating, registration, and call setup. In particular, each base station maintains a database of mobile stations currently under its jurisdiction. Information needed to maintain this database is sent on the dedicated control channel.

Finally, there is the **common control channel**, which is split up into three logical subchannels. The first of these subchannels is the **paging channel**, which the base station uses to announce incoming calls. Each mobile station monitors it continuously to watch for calls it should answer. The second is the **random access channel**, which allows users to request a slot on the dedicated control channel. If two requests collide, they are garbled and have to be retried later. Using the dedicated control channel slot, the station can set up a call. The assigned slot is announced on the third subchannel, the **access grant channel**.

CDMA—Code Division Multiple Access

D-AMPS and GSM are fairly conventional systems. They use both FDM and TDM to divide the spectrum into channels and the channels into time slots. However, there is a third kid on the block, **CDMA** (**Code Division Multiple Access**), which works completely differently. When CDMA was first proposed, the industry gave it approximately the same reaction that Columbus first got from Queen Isabella when he proposed reaching India by sailing in the wrong direction. However, through the persistence of a single company, Qualcomm, CDMA has matured to the point where it is not only acceptable, it is now viewed as the best technical solution around and the basis for the third-generation mobile systems. It is also widely used in the U.S. in second-generation mobile systems, competing head-on with D-AMPS. For example, Sprint PCS uses CDMA, whereas AT&T Wireless uses D-AMPS. CDMA is described in International Standard IS-95 and is sometimes referred to by that name. The brand name **cdmaOne** is also used.

CDMA is completely different from AMPS, D-AMPS, and GSM. Instead of dividing the allowed frequency range into a few hundred narrow channels, CDMA allows each station to transmit over the entire frequency spectrum all the time. Multiple simultaneous transmissions are separated using coding theory. CDMA also relaxes the assumption that colliding frames are totally garbled. Instead, it assumes that multiple signals add linearly.

Before getting into the algorithm, let us consider an analogy: an airport lounge with many pairs of people conversing. TDM is comparable to all the people being in the middle of the room but taking turns speaking. FDM is comparable to the people being in widely separated clumps, each clump holding its own conversation at the same time as, but still independent of, the others. CDMA is comparable to everybody being in the middle of the room talking at once, but with each pair in a different language. The French-speaking couple just hones in on the French, rejecting everything that is not French as noise. Thus, the key to CDMA is to be able to extract the desired signal while rejecting everything else as random noise. A somewhat simplified description of CDMA follows.

In CDMA, each bit time is subdivided into m short intervals called **chips**. Typically, there are 64 or 128 chips per bit, but in the example given below we will use 8 chips/bit for simplicity.

Each station is assigned a unique m-bit code called a **chip sequence**. To transmit a 1 bit, a station sends its chip sequence. To transmit a 0 bit, it sends the one's complement of its chip sequence. No other patterns are permitted. Thus, for $m = 8$, if station A is assigned the chip sequence 00011011, it sends a 1 bit by sending 00011011 and a 0 bit by sending 11100100.

Increasing the amount of information to be sent from b bits/sec to mb chips/sec can only be done if the bandwidth available is increased by a factor of m, making CDMA a form of spread spectrum communication (assuming no changes in the modulation or encoding techniques). If we have a 1-MHz band available for 100 stations, with FDM each one would have 10 kHz and could send at 10 kbps (assuming 1 bit per Hz). With CDMA, each station uses the full 1 MHz, so the chip rate is 1 megachip per second. With fewer than 100 chips per bit, the effective bandwidth per station is higher for CDMA than FDM, and the channel allocation problem is also solved.

For pedagogical purposes, it is more convenient to use a bipolar notation, with binary 0 being −1 and binary 1 being +1. We will show chip sequences in parentheses, so a 1 bit for station A now becomes (−1 −1 −1 +1 +1 −1 +1 +1). In Fig. 2-45(a) we show the binary chip sequences assigned to four example stations. In Fig. 2-45(b) we show them in our bipolar notation.

(a)	(b)
A: 0 0 0 1 1 0 1 1	A: (−1 −1 −1 +1 +1 −1 +1 +1)
B: 0 0 1 0 1 1 1 0	B: (−1 −1 +1 −1 +1 +1 +1 −1)
C: 0 1 0 1 1 1 0 0	C: (−1 +1 −1 +1 +1 +1 −1 −1)
D: 0 1 0 0 0 0 1 0	D: (−1 +1 −1 −1 −1 −1 +1 −1)

Six examples:

– – 1 –	**C**	S_1 = (−1 +1 −1 +1 +1 +1 −1 −1)
– 1 1 –	**B + C**	S_2 = (−2 0 0 0 +2 +2 0 −2)
1 0 – –	**A +** $\bar{\mathbf{B}}$	S_3 = (0 0 −2 +2 0 −2 0 +2)
1 0 1 –	**A +** $\bar{\mathbf{B}}$ **+ C**	S_4 = (−1 +1 −3 +3 +1 −1 −1 +1)
1 1 1 1	**A + B + C + D**	S_5 = (−4 0 −2 0 +2 0 +2 −2)
1 1 0 1	**A + B +** $\bar{\mathbf{C}}$ **+ D**	S_6 = (−2 −2 0 −2 0 −2 +4 0)

(c)

$S_1 \bullet C = (1 +1 +1 +1 +1 +1 +1 +1)/8 = 1$
$S_2 \bullet C = (2 +0 +0 +0 +2 +2 +0 +2)/8 = 1$
$S_3 \bullet C = (0 +0 +2 +2 +0 -2 +0 -2)/8 = 0$
$S_4 \bullet C = (1 +1 +3 +3 +1 -1 +1 -1)/8 = 1$
$S_5 \bullet C = (4 +0 +2 +0 +2 +0 -2 +2)/8 = 1$
$S_6 \bullet C = (2 -2 +0 -2 +0 -2 -4 +0)/8 = -1$

(d)

Figure 2-45. (a) Binary chip sequences for four stations. (b) Bipolar chip sequences. (c) Six examples of transmissions. (d) Recovery of station C's signal.

Each station has its own unique chip sequence. Let us use the symbol **S** to indicate the m-chip vector for station S, and $\bar{\mathbf{S}}$ for its negation. All chip sequences are pairwise **orthogonal**, by which we mean that the normalized inner product of any two distinct chip sequences, **S** and **T** (written as **S•T**), is 0. It is known how to generate such orthogonal chip sequences using a method known as **Walsh codes**. In mathematical terms, orthogonality of the chip sequences can be expressed as follows:

$$\mathbf{S}\bullet\mathbf{T} \equiv \frac{1}{m}\sum_{i=1}^{m} S_i T_i = 0 \tag{2-4}$$

In plain English, as many pairs are the same as are different. This orthogonality property will prove crucial later on. Note that if $\mathbf{S}\bullet\mathbf{T} = 0$, then $\mathbf{S}\bullet\overline{\mathbf{T}}$ is also 0. The normalized inner product of any chip sequence with itself is 1:

$$\mathbf{S}\bullet\mathbf{S} = \frac{1}{m}\sum_{i=1}^{m} S_i S_i = \frac{1}{m}\sum_{i=1}^{m} S_i^2 = \frac{1}{m}\sum_{i=1}^{m} (\pm 1)^2 = 1$$

This follows because each of the m terms in the inner product is 1, so the sum is m. Also note that $\mathbf{S}\bullet\overline{\mathbf{S}} = -1$.

During each bit time, a station can transmit a 1 by sending its chip sequence, it can transmit a 0 by sending the negative of its chip sequence, or it can be silent and transmit nothing. For the moment, we assume that all stations are synchronized in time, so all chip sequences begin at the same instant.

When two or more stations transmit simultaneously, their bipolar signals add linearly. For example, if in one chip period three stations output +1 and one station outputs −1, the result is +2. One can think of this as adding voltages: three stations outputting +1 volts and 1 station outputting −1 volts gives 2 volts.

In Fig. 2-45(c) we see six examples of one or more stations transmitting at the same time. In the first example, C transmits a 1 bit, so we just get C's chip sequence. In the second example, both B and C transmit 1 bits, so we get the sum of their bipolar chip sequences, namely:

$$(-1\ -1\ +1\ -1\ +1\ +1\ +1\ -1) + (-1\ +1\ -1\ +1\ +1\ +1\ -1\ -1) = (-2\ \ 0\ \ 0\ \ 0\ +2\ +2\ \ 0\ -2)$$

In the third example, station A sends a 1 and station B sends a 0. The others are silent. In the fourth example, A and C send a 1 bit while B sends a 0 bit. In the fifth example, all four stations send a 1 bit. Finally, in the last example, A, B, and D send a 1 bit, while C sends a 0 bit. Note that each of the six sequences S_1 through S_6 given in Fig. 2-45(c) represents only one bit time.

To recover the bit stream of an individual station, the receiver must know that station's chip sequence in advance. It does the recovery by computing the normalized inner product of the received chip sequence (the linear sum of all the stations that transmitted) and the chip sequence of the station whose bit stream it is trying to recover. If the received chip sequence is **S** and the receiver is trying to listen to a station whose chip sequence is **C**, it just computes the normalized inner product, **S•C**.

To see why this works, just imagine that two stations, A and C, both transmit a 1 bit at the same time that B transmits a 0 bit. The receiver sees the sum, $\mathbf{S} = \mathbf{A} + \overline{\mathbf{B}} + \mathbf{C}$ and computes

$$\mathbf{S}\bullet\mathbf{C} = (\mathbf{A} + \overline{\mathbf{B}} + \mathbf{C})\bullet\mathbf{C} = \mathbf{A}\bullet\mathbf{C} + \overline{\mathbf{B}}\bullet\mathbf{C} + \mathbf{C}\bullet\mathbf{C} = 0 + 0 + 1 = 1$$

The first two terms vanish because all pairs of chip sequences have been carefully chosen to be orthogonal, as shown in Eq. (2-4). Now it should be clear why this property must be imposed on the chip sequences.

An alternative way of thinking about this situation is to imagine that the three chip sequences all came in separately, rather than summed. Then, the receiver would compute the inner product with each one separately and add the results. Due to the orthogonality property, all the inner products except $\mathbf{C}\bullet\mathbf{C}$ would be 0. Adding them and then doing the inner product is in fact the same as doing the inner products and then adding those.

To make the decoding process more concrete, let us consider the six examples of Fig. 2-45(c) again as illustrated in Fig. 2-45(d). Suppose that the receiver is interested in extracting the bit sent by station C from each of the six sums S_1 through S_6. It calculates the bit by summing the pairwise products of the received $\mathbf{S}$ and the $\mathbf{C}$ vector of Fig. 2-45(b) and then taking 1/8 of the result (since $m = 8$ here). As shown, the correct bit is decoded each time. It is just like speaking French.

In an ideal, noiseless CDMA system, the capacity (i.e., number of stations) can be made arbitrarily large, just as the capacity of a noiseless Nyquist channel can be made arbitrarily large by using more and more bits per sample. In practice, physical limitations reduce the capacity considerably. First, we have assumed that all the chips are synchronized in time. In reality, such synchronization is impossible. What can be done is that the sender and receiver synchronize by having the sender transmit a predefined chip sequence that is long enough for the receiver to lock onto. All the other (unsynchronized) transmissions are then seen as random noise. If there are not too many of them, however, the basic decoding algorithm still works fairly well. A large body of theory exists relating the superposition of chip sequences to noise level (Pickholtz et al., 1982). As one might expect, the longer the chip sequence, the higher the probability of detecting it correctly in the presence of noise. For extra reliability, the bit sequence can use an error-correcting code. Chip sequences never use error-correcting codes.

An implicit assumption in our discussion is that the power levels of all stations are the same as perceived by the receiver. CDMA is typically used for wireless systems with a fixed base station and many mobile stations at varying distances from it. The power levels received at the base station depend on how far away the transmitters are. A good heuristic here is for each mobile station to transmit to the base station at the inverse of the power level it receives from the base station. In other words, a mobile station receiving a weak signal from the will use more power than one getting a strong signal. The base station can also give explicit commands to the mobile stations to increase or decrease their transmission power.

We have also assumed that the receiver knows who the sender is. In principle, given enough computing capacity, the receiver can listen to all the senders at once by running the decoding algorithm for each of them in parallel. In real life, suffice it to say that this is easier said than done. CDMA also has many other complicating factors that have been glossed over in this brief introduction. Nevertheless, CDMA is a clever scheme that is being rapidly introduced for wireless mobile communication. It normally operates in a band of 1.25 MHz (versus 30 kHz for D-AMPS and 200 kHz for GSM), but it supports many more users in that band than either of the other systems. In practice, the bandwidth available to each user is at least as good as GSM and often much better.

Engineers who want to gain a very deep understanding of CDMA should read (Lee and Miller, 1998). An alternative spreading scheme, in which the spreading is over time rather than frequency, is described in (Crespo et al., 1995). Yet another scheme is described in (Sari et al., 2000). All of these references require quite a bit of background in communication engineering.

2.6.3 Third-Generation Mobile Phones: Digital Voice and Data

What is the future of mobile telephony? Let us take a quick look. A number of factors are driving the industry. First, data traffic already exceeds voice traffic on the fixed network and is growing exponentially, whereas voice traffic is essentially flat. Many industry experts expect data traffic to dominate voice on mobile devices as well soon. Second, the telephone, entertainment, and computer industries have all gone digital and are rapidly converging. Many people are drooling over a lightweight, portable device that acts as a telephone, CD player, DVD player, e-mail terminal, Web interface, gaming machine, word processor, and more, all with worldwide wireless connectivity to the Internet at high bandwidth. This device and how to connect it is what third generation mobile telephony is all about. For more information, see (Huber et al., 2000; and Sarikaya, 2000).

Back in 1992, ITU tried to get a bit more specific about this dream and issued a blueprint for getting there called **IMT-2000**, where IMT stood for **International Mobile Telecommunications**. The number 2000 stood for three things: (1) the year it was supposed to go into service, (2) the frequency it was supposed to operate at (in MHz), and (3) the bandwidth the service should have (in kHz).

It did not make it on any of the three counts. Nothing was implemented by 2000. ITU recommended that all governments reserve spectrum at 2 GHz so devices could roam seamlessly from country to country. China reserved the required bandwidth but nobody else did. Finally, it was recognized that 2 Mbps is not currently feasible for users who are *too* mobile (due to the difficulty of performing handoffs quickly enough). More realistic is 2 Mbps for stationary indoor users (which will compete head-on with ADSL), 384 kbps for people walking, and 144 kbps for connections in cars. Nevertheless, the whole area of **3G**, as it is called, is one great cauldron of activity. The third generation may be a bit less than originally hoped for and a bit late, but it will surely happen.

The basic services that the IMT-2000 network is supposed to provide to its users are:

1. High-quality voice transmission.
2. Messaging (replacing e-mail, fax, SMS, chat, etc.).
3. Multimedia (playing music, viewing videos, films, television, etc.).
4. Internet access (Web surfing, including pages with audio and video).

Additional services might be video conferencing, telepresence, group game playing, and m-commerce (waving your telephone at the cashier to pay in a store). Furthermore, all these services are supposed to be available worldwide (with automatic connection via a satellite when no terrestrial network can be located), instantly (always on), and with quality-of-service guarantees.

ITU envisioned a single worldwide technology for IMT-2000, so that manufacturers could build a single device that could be sold and used anywhere in the world (like CD players and computers and unlike mobile phones and televisions). Having a single technology would also make life much simpler for network operators and would encourage more people to use the services. Format wars, such as the Betamax versus VHS battle when videorecorders first came out, are not good for business.

Several proposals were made, and after some winnowing, it came down to two main ones. The first one, **W-CDMA** (**Wideband CDMA**), was proposed by Ericsson. This system uses direct sequence spread spectrum of the type we described above. It runs in a 5 MHz bandwidth and has been designed to interwork with GSM networks although it is not backward compatible with GSM. It does, however, have the property that a caller can leave a W-CDMA cell and enter a GSM cell without losing the call. This system was pushed hard by the European Union, which called it **UMTS** (**Universal Mobile Telecommunications System**).

The other contender was **CDMA2000**, proposed by Qualcomm. It, too, is a direct sequence spread spectrum design, basically an extension of IS-95 and backward compatible with it. It also uses a 5-MHz bandwidth, but it has not been designed to interwork with GSM and cannot hand off calls to a GSM cell (or a D-AMPS cell, for that matter). Other technical differences with W-CDMA include a different chip rate, different frame time, different spectrum used, and a different way to do time synchronization.

If the Ericsson and Qualcomm engineers were put in a room and told to come to a common design, they probably could. After all, the basic principle behind both systems is CDMA in a 5 MHz channel and nobody is willing to die for his preferred chip rate. The trouble is that the real problem is not engineering, but politics (as usual). Europe wanted a system that interworked with GSM; the U.S. wanted a system that was compatible with one already widely deployed in the U.S. (IS-95). Each side also supported its local company (Ericsson is based in Sweden; Qualcomm is in California). Finally, Ericsson and Qualcomm were involved in numerous lawsuits over their respective CDMA patents.

In March 1999, the two companies settled the lawsuits when Ericsson agreed to buy Qualcomm's infrastructure. They also agreed to a single 3G standard, but one with multiple incompatible options, which to a large extent just papers over the technical differences. These disputes notwithstanding, 3G devices and services are likely to start appearing in the coming years.

Much has been written about 3G systems, most of it praising it as the greatest thing since sliced bread. Some references are (Collins and Smith, 2001; De Vriendt et al., 2002; Harte et al., 2002; Lu, 2002; and Sarikaya, 2000). However, some dissenters think that the industry is pointed in the wrong direction (Garber, 2002; and Goodman, 2000).

While waiting for the fighting over 3G to stop, some operators are gingerly taking a cautious small step in the direction of 3G by going to what is sometimes called **2.5G**, although 2.1G might be more accurate. One such system is **EDGE** (**Enhanced Data rates for GSM Evolution**), which is just GSM with more bits per baud. The trouble is, more bits per baud also means more errors per baud, so EDGE has nine different schemes for modulation and error correction, differing on how much of the bandwidth is devoted to fixing the errors introduced by the higher speed.

Another 2.5G scheme is **GPRS** (**General Packet Radio Service**), which is an overlay packet network on top of D-AMPS or GSM. It allows mobile stations to send and receive IP packets in a cell running a voice system. When GPRS is in operation, some time slots on some frequencies are reserved for packet traffic. The number and location of the time slots can be dynamically managed by the base station, depending on the ratio of voice to data traffic in the cell.

The available time slots are divided into several logical channels, used for different purposes. The base station determines which logical channels are mapped onto which time slots. One logical channel is for downloading packets from the base station to some mobile station, with each packet indicating who it is destined for. To send an IP packet, a mobile station requests one or more time slots by sending a request to the base station. If the request arrives without damage, the base station announces the frequency and time slots allocated to the mobile for sending the packet. Once the packet has arrived at the base station, it is transferred to the Internet by a wired connection. Since GPRS is just an overlay over the existing voice system, it is at best a stop-gap measure until 3G arrives.

Even though 3G networks are not fully deployed yet, some researchers regard 3G as a done deal and thus not interesting any more. These people are already working on 4G systems (Berezdivin et al., 2002; Guo and Chaskar, 2002; Huang and Zhuang, 2002; Kellerer et al., 2002; and Misra et al., 2002). Some of the proposed features of 4G systems include high bandwidth, ubiquity (connectivity everywhere), seamless integration with wired networks and especially IP, adaptive resource and spectrum management, software radios, and high quality of service for multimedia.

Then on the other hand, so many 802.11 wireless LAN access points are being set up all over the place, that some people think 3G is not only not a done deal, it is doomed. In this vision, people will just wander from one 802.11 access point to another to stay connected. To say the industry is in a state of enormous flux is a huge understatement. Check back in about 5 years to see what happens.

2.7 CABLE TELEVISION

We have now studied both the fixed and wireless telephone systems in a fair amount of detail. Both will clearly play a major role in future networks. However, an alternative available for fixed networking is now becoming a major player: cable television networks. Many people already get their telephone and Internet service over the cable, and the cable operators are actively working to increase their market share. In the following sections we will look at cable television as a networking system in more detail and contrast it with the telephone systems we have just studied. For more information about cable, see (Laubach et al., 2001; Louis, 2002; Ovadia, 2001; and Smith, 2002).

2.7.1 Community Antenna Television

Cable television was conceived in the late 1940s as a way to provide better reception to people living in rural or mountainous areas. The system initially consisted of a big antenna on top of a hill to pluck the television signal out of the air, an amplifier, called the **head end**, to strengthen it, and a coaxial cable to deliver it to people's houses, as illustrated in Fig. 2-46.

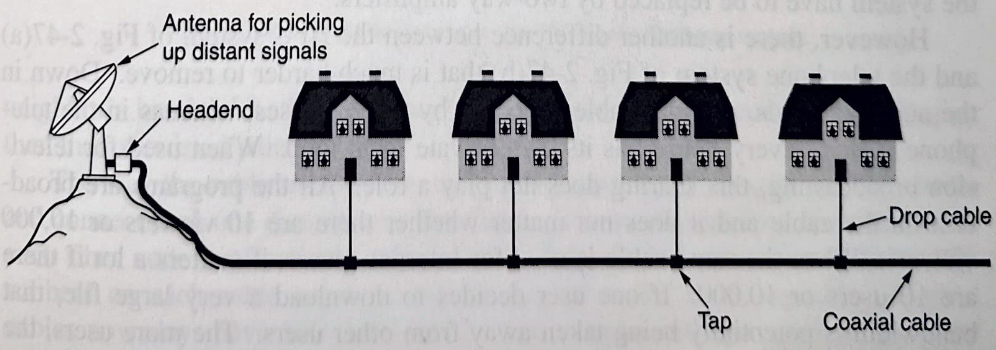

Figure 2-46. An early cable television system.

time from the earth to the satellite is 270 msec. ACK frames never occur. NAK frames are 40 bits. The error rate for data frames is 1 percent, and the error rate for NAK frames is negligible. The sequence numbers are 8 bits.

31. Consider an error-free 64-kbps satellite channel used to send 512-byte data frames in one direction, with very short acknowledgements coming back the other way. What is the maximum throughput for window sizes of 1, 7, 15, and 127? The earth-satellite propagation time is 270 msec.

32. A 100-km-long cable runs at the T1 data rate. The propagation speed in the cable is 2/3 the speed of light in vacuum. How many bits fit in the cable?

33. Suppose that we model protocol 4 using the finite state machine model. How many states exist for each machine? How many states exist for the communication channel? How many states exist for the complete system (two machines and the channel)? Ignore the checksum errors.

34. Give the firing sequence for the Petri net of Fig. 3-23 corresponding to the state sequence (000), (01A), (01—), (010), (01A) in Fig. 3-21. Explain in words what the sequence represents.

35. Given the transition rules $AC \rightarrow B$, $B \rightarrow AC$, $CD \rightarrow E$, and $E \rightarrow CD$, draw the Petri net described. From the Petri net, draw the finite state graph reachable from the initial state ACD. What well-known concept do these transition rules model?

36. PPP is based closely on HDLC, which uses bit stuffing to prevent accidental flag bytes within the payload from causing confusion. Give at least one reason why PPP uses byte stuffing instead.

37. What is the minimum overhead to send an IP packet using PPP? Count only the overhead introduced by PPP itself, not the IP header overhead.

38. The goal of this lab exercise is to implement an error detection mechanism using the standard CRC algorithm described in the text. Write two programs, generator and verifier. The generator program reads from standard input an n-bit message as a string of 0s and 1s as a line of ASCII text. The second line is the k-bit polynomial, also in ASCII. It outputs to standard output a line of ASCII text with $n + k$ 0s and 1s representing the message to be transmitted. Then it outputs the polynomial, just as it read it in. The verifier program reads in the output of the generator program and outputs a message indicating whether it is correct or not. Finally, write a program, alter, that inverts one bit on the first line depending on its argument (the bit number counting the leftmost bit as 1) but copies the rest of the two lines correctly. By typing:

```
generator <file | verifier
```

you should see that the message is correct, but by typing

```
generator <file | alter arg | verifier
```

you should get the error message.

39. Write a program to simulate the behavior of a Petri net. The program should read in the transition rules as well as a list of states corresponding to the network link layer issuing a new packet or accepting a new packet. From the initial state, also read in, the program should pick enabled transitions at random and fire them, checking to see if a host ever accepts 2 packets without the other host emitting a new one in between.

4

THE MEDIUM ACCESS CONTROL SUBLAYER

As we pointed out in Chap. 1, networks can be divided into two categories: those using point-to-point connections and those using broadcast channels. This chapter deals with broadcast networks and their protocols.

In any broadcast network, the key issue is how to determine who gets to use the channel when there is competition for it. To make this point clearer, consider a conference call in which six people, on six different telephones, are all connected so that each one can hear and talk to all the others. It is very likely that when one of them stops speaking, two or more will start talking at once, leading to chaos. In a face-to-face meeting, chaos is avoided by external means, for example, at a meeting, people raise their hands to request permission to speak. When only a single channel is available, determining who should go next is much harder. Many protocols for solving the problem are known and form the contents of this chapter. In the literature, broadcast channels are sometimes referred to as **multiaccess channels** or **random access channels**.

The protocols used to determine who goes next on a multiaccess channel belong to a sublayer of the data link layer called the **MAC (Medium Access Control)** sublayer. The MAC sublayer is especially important in LANs, many of which use a multiaccess channel as the basis for communication. WANs, in contrast, use point-to-point links, except for satellite networks. Because multiaccess channels and LANs are so closely related, in this chapter we will discuss LANs in general, including a few issues that are not strictly part of the MAC sublayer.

Technically, the MAC sublayer is the bottom part of the data link layer, so logically we should have studied it before examining all the point-to-point protocols in Chap. 3. Nevertheless, for most people, understanding protocols involving multiple parties is easier after two-party protocols are well understood. For that reason we have deviated slightly from a strict bottom-up order of presentation.

4.1 THE CHANNEL ALLOCATION PROBLEM

The central theme of this chapter is how to allocate a single broadcast channel among competing users. We will first look at static and dynamic schemes in general. Then we will examine a number of specific algorithms.

4.1.1 Static Channel Allocation in LANs and MANs

The traditional way of allocating a single channel, such as a telephone trunk, among multiple competing users is Frequency Division Multiplexing (FDM). If there are N users, the bandwidth is divided into N equal-sized portions (see Fig. 2-31), each user being assigned one portion. Since each user has a private frequency band, there is no interference between users. When there is only a small and constant number of users, each of which has a heavy (buffered) load of traffic (e.g., carriers' switching offices), FDM is a simple and efficient allocation mechanism.

However, when the number of senders is large and continuously varying or the traffic is bursty, FDM presents some problems. If the spectrum is cut up into N regions and fewer than N users are currently interested in communicating, a large piece of valuable spectrum will be wasted. If more than N users want to communicate, some of them will be denied permission for lack of bandwidth, even if some of the users who have been assigned a frequency band hardly ever transmit or receive anything.

However, even assuming that the number of users could somehow be held constant at N, dividing the single available channel into static subchannels is inherently inefficient. The basic problem is that when some users are quiescent, their bandwidth is simply lost. They are not using it, and no one else is allowed to use it either. Furthermore, in most computer systems, data traffic is extremely bursty (peak traffic to mean traffic ratios of 1000:1 are common). Consequently, most of the channels will be idle most of the time.

The poor performance of static FDM can easily be seen from a simple queueing theory calculation. Let us start with the mean time delay, T, for a channel of capacity C bps, with an arrival rate of λ frames/sec, each frame having a length drawn from an exponential probability density function with mean $1/\mu$ bits/frame.

With these parameters the arrival rate is λ frames/sec and the service rate is μC frames/sec. From queueing theory it can be shown that for Poisson arrival and service times,

$$T = \frac{1}{\mu C - \lambda}$$

For example, if C is 100 Mbps, the mean frame length, $1/\mu$, is 10,000 bits, and the frame arrival rate, λ, is 5000 frames/sec, then $T = 200\ \mu$sec. Note that if we ignored the queueing delay and just asked how long it takes to send a 10,000 bit frame on a 100-Mbps network, we would get the (incorrect) answer of 100 μsec. That result only holds when there is no contention for the channel.

Now let us divide the single channel into N independent subchannels, each with capacity C/N bps. The mean input rate on each of the subchannels will now be λ/N. Recomputing T we get

$$T_{\mathrm{FDM}} = \frac{1}{\mu(C/N) - (\lambda/N)} = \frac{N}{\mu C - \lambda} = NT \qquad (4\text{-}1)$$

The mean delay using FDM is N times worse than if all the frames were somehow magically arranged orderly in a big central queue.

Precisely the same arguments that apply to FDM also apply to time division multiplexing (TDM). Each user is statically allocated every Nth time slot. If a user does not use the allocated slot, it just lies fallow. The same holds if we split up the networks physically. Using our previous example again, if we were to replace the 100-Mbps network with 10 networks of 10 Mbps each and statically allocate each user to one of them, the mean delay would jump from 200 μsec to 2 msec.

Since none of the traditional static channel allocation methods work well with bursty traffic, we will now explore dynamic methods.

4.1.2 Dynamic Channel Allocation in LANs and MANs

Before we get into the first of the many channel allocation methods to be discussed in this chapter, it is worthwhile carefully formulating the allocation problem. Underlying all the work done in this area are five key assumptions, described below.

1. **Station Model**. The model consists of N independent **stations** (e.g., computers, telephones, or personal communicators), each with a program or user that generates frames for transmission. Stations are sometimes called **terminals**. The probability of a frame being generated in an interval of length Δt is $\lambda \Delta t$, where λ is a constant (the arrival rate of new frames). Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.

2. **Single Channel Assumption**. A single channel is available for all communication. All stations can transmit on it and all can receive from it. As far as the hardware is concerned, all stations are equivalent, although protocol software may assign priorities to them.

3. **Collision Assumption**. If two frames are transmitted simultaneously, they overlap in time and the resulting signal is garbled. This event is called a **collision**. All stations can detect collisions. A collided frame must be transmitted again later. There are no errors other than those generated by collisions.

4a. **Continuous Time**. Frame transmission can begin at any instant. There is no master clock dividing time into discrete intervals.

4b. **Slotted Time**. Time is divided into discrete intervals (slots). Frame transmissions always begin at the start of a slot. A slot may contain 0, 1, or more frames, corresponding to an idle slot, a successful transmission, or a collision, respectively.

5a. **Carrier Sense**. Stations can tell if the channel is in use before trying to use it. If the channel is sensed as busy, no station will attempt to use it until it goes idle.

5b. **No Carrier Sense**. Stations cannot sense the channel before trying to use it. They just go ahead and transmit. Only later can they determine whether the transmission was successful.

Some discussion of these assumptions is in order. The first one says that stations are independent and that work is generated at a constant rate. It also implicitly assumes that each station only has one program or user, so while the station is blocked, no new work is generated. More sophisticated models allow multiprogrammed stations that can generate work while a station is blocked, but the analysis of these stations is much more complex.

The single channel assumption is the heart of the model. There are no external ways to communicate. Stations cannot raise their hands to request that the teacher call on them.

The collision assumption is also basic, although in some systems (notably spread spectrum), this assumption is relaxed, with surprising results. Also, some LANs, such as token rings, pass a special token from station to station, possession of which allows the current holder to transmit a frame. But in the coming sections we will stick to the single channel with contention and collisions model.

Two alternative assumptions about time are possible. Either it is continuous (4a) or it is slotted (4b). Some systems use one and some systems use the other, so we will discuss and analyze both. For a given system, only one of them holds.

Similarly, a network can either have carrier sensing (5a) or not have it (5b). LANs generally have carrier sense. However, wireless networks cannot use it effectively because not every station may be within radio range of every other station. Stations on wired carrier sense networks can terminate their transmission prematurely if they discover that it is colliding with another transmission. Collision detection is rarely done on wireless networks, for engineering reasons. Note that the word "carrier" in this sense refers to an electrical signal on the cable and has nothing to do with the common carriers (e.g., telephone companies) that date back to the Pony Express days.

4.2 MULTIPLE ACCESS PROTOCOLS

Many algorithms for allocating a multiple access channel are known. In the following sections we will study a small sample of the more interesting ones and give some examples of their use.

4.2.1 ALOHA

In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. Their work has been extended by many researchers since then (Abramson, 1985). Although Abramson's work, called the ALOHA system, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel.

We will discuss two versions of ALOHA here: pure and slotted. They differ with respect to whether time is divided into discrete slots into which all frames must fit. Pure ALOHA does not require global time synchronization; slotted ALOHA does.

Pure ALOHA

The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. However, due to the feedback property of broadcasting, a sender can always find out whether its frame was destroyed by listening to the channel, the same way other users do. With a LAN, the feedback is immediate; with a satellite, there is a delay of 270 msec before the sender knows if the transmission was successful. If listening while transmitting is not possible for some reason, acknowledgements are needed. If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are widely known as **contention** systems.

A sketch of frame generation in an ALOHA system is given in Fig. 4-1. We have made the frames all the same length because the throughput of ALOHA systems is maximized by having a uniform frame size rather than by allowing variable length frames.

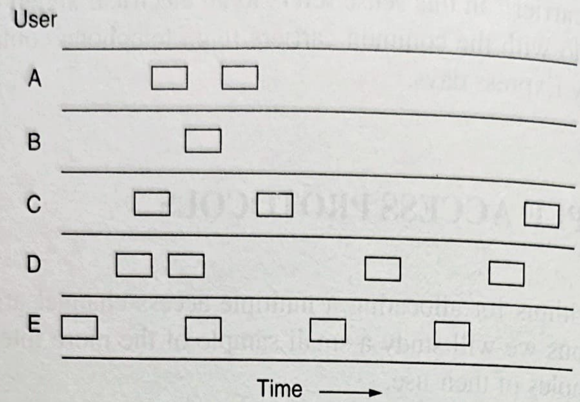

Figure 4-1. In pure ALOHA, frames are transmitted at completely arbitrary times.

Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted later. The checksum cannot (and should not) distinguish between a total loss and a near miss. Bad is bad.

An interesting question is: What is the efficiency of an ALOHA channel? In other words, what fraction of all transmitted frames escape collisions under these chaotic circumstances? Let us first consider an infinite collection of interactive users sitting at their computers (stations). A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line and checks the channel to see if it was successful. If so, the user sees the reply and goes back to typing. If not, the user continues to wait and the frame is retransmitted over and over until it has been successfully sent.

Let the "frame time" denote the amount of time needed to transmit the standard, fixed-length frame (i.e., the frame length divided by the bit rate). At this point we assume that the infinite population of users generates new frames according to a Poisson distribution with mean N frames per frame time. (The infinite-population assumption is needed to ensure that N does not decrease as users become blocked.) If $N > 1$, the user community is generating frames at a higher rate than the channel can handle, and nearly every frame will suffer a collision. For reasonable throughput we would expect $0 < N < 1$.

In addition to the new frames, the stations also generate retransmissions of frames that previously suffered collisions. Let us further assume that the probability of k transmission attempts per frame time, old and new combined, is also

Poisson, with mean G per frame time. Clearly, $G \geq N$. At low load (i.e., $N \approx 0$), there will be few collisions, hence few retransmissions, so $G \approx N$. At high load there will be many collisions, so $G > N$. Under all loads, the throughput, S, is just the offered load, G, times the probability, P_0, of a transmission succeeding—that is, $S = GP_0$, where P_0 is the probability that a frame does not suffer a collision.

A frame will not suffer a collision if no other frames are sent within one frame time of its start, as shown in Fig. 4-2. Under what conditions will the shaded frame arrive undamaged? Let t be the time required to send a frame. If any other user has generated a frame between time t_0 and $t_0 + t$, the end of that frame will collide with the beginning of the shaded one. In fact, the shaded frame's fate was already sealed even before the first bit was sent, but since in pure ALOHA a station does not listen to the channel before transmitting, it has no way of knowing that another frame was already underway. Similarly, any other frame started between $t_0 + t$ and $t_0 + 2t$ will bump into the end of the shaded frame.

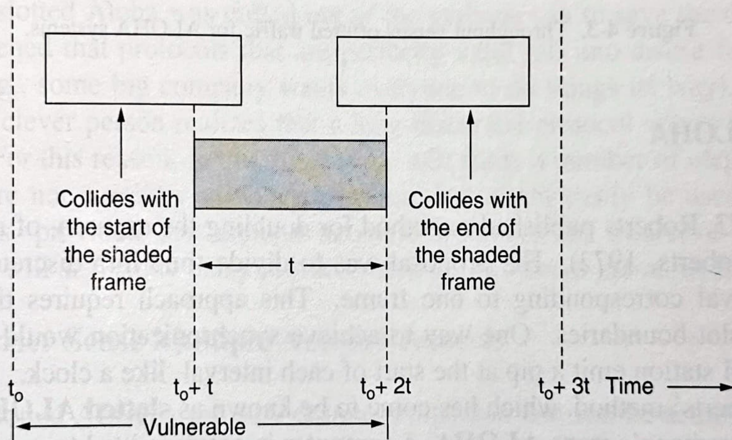

Figure 4-2. Vulnerable period for the shaded frame.

The probability that k frames are generated during a given frame time is given by the Poisson distribution:

$$\Pr[k] = \frac{G^k e^{-G}}{k!} \tag{4-2}$$

so the probability of zero frames is just e^{-G}. In an interval two frame times long, the mean number of frames generated is $2G$. The probability of no other traffic being initiated during the entire vulnerable period is thus given by $P_0 = e^{-2G}$. Using $S = GP_0$, we get

$$S = Ge^{-2G}$$

The relation between the offered traffic and the throughput is shown in Fig. 4-3. The maximum throughput occurs at $G = 0.5$, with $S = 1/2e$, which is about 0.184. In other words, the best we can hope for is a channel utilization of

18 percent. This result is not very encouraging, but with everyone transmitting at will, we could hardly have expected a 100 percent success rate.

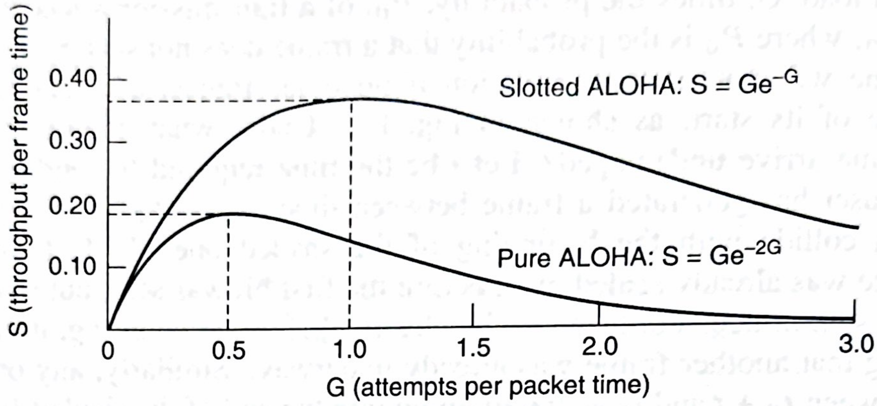

Figure 4-3. Throughput versus offered traffic for ALOHA systems.

Slotted ALOHA

In 1972, Roberts published a method for doubling the capacity of an ALOHA system (Roberts, 1972). His proposal was to divide time into discrete intervals, each interval corresponding to one frame. This approach requires the users to agree on slot boundaries. One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.

In Roberts' method, which has come to be known as **slotted ALOHA**, in contrast to Abramson's **pure ALOHA**, a computer is not permitted to send whenever a carriage return is typed. Instead, it is required to wait for the beginning of the next slot. Thus, the continuous pure ALOHA is turned into a discrete one. Since the vulnerable period is now halved, the probability of no other traffic during the same slot as our test frame is e^{-G} which leads to

$$S = Ge^{-G} \tag{4-3}$$

As you can see from Fig. 4-3, slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G = 1$, the probability of an empty slot is 0.368 (from Eq. 4-2). The best we can hope for using slotted ALOHA is 37 percent of the slots empty, 37 percent successes, and 26 percent collisions. Operating at higher values of G reduces the number of empties but increases the number of collisions exponentially. To see how this rapid growth of collisions with G comes about, consider the transmission of a test frame. The probability that it will avoid a collision is e^{-G}, the probability that all the other users are silent in that slot. The probability of a collision is then just $1 - e^{-G}$. The probability of a transmission requiring exactly k attempts, (i.e., $k - 1$ collisions followed by one success) is

$$P_k = e^{-G}(1 - e^{-G})^{k-1}$$

The expected number of transmissions, E, per carriage return typed is then

$$E = \sum_{k=1}^{\infty} kP_k = \sum_{k=1}^{\infty} ke^{-G}(1 - e^{-G})^{k-1} = e^G$$

As a result of the exponential dependence of E upon G, small increases in the channel load can drastically reduce its performance.

Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. When Internet access over the cable was invented, all of a sudden there was a problem of how to allocate a shared channel among multiple competing users, and slotted Aloha was pulled out of the garbage can to save the day. It has often happened that protocols that are perfectly valid fall into disuse for political reasons (e.g., some big company wants everyone to do things its way), but years later some clever person realizes that a long-discarded protocol solves his current problem. For this reason, in this chapter we will study a number of elegant protocols that are not currently in widespread use, but might easily be used in future applications, provided that enough network designers are aware of them. Of course, we will also study many protocols that are in current use as well.

4.2.2 Carrier Sense Multiple Access Protocols

With slotted ALOHA the best channel utilization that can be achieved is $1/e$. This is hardly surprising, since with stations transmitting at will, without paying attention to what the other stations are doing, there are bound to be many collisions. In local area networks, however, it is possible for stations to detect what other stations are doing, and adapt their behavior accordingly. These networks can achieve a much better utilization than $1/e$. In this section we will discuss some protocols for improving performance.

Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called **carrier sense protocols**. A number of them have been proposed. Kleinrock and Tobagi (1975) have analyzed several such protocols in detail. Below we will mention several versions of the carrier sense protocols.

Persistent and Nonpersistent CSMA

The first carrier sense protocol that we will study here is called **1-persistent CSMA** (Carrier Sense Multiple Access). When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 when it finds the channel idle.

The propagation delay has an important effect on the performance of the protocol. There is a small chance that just after a station begins sending, another station will become ready to send and sense the channel. If the first station's signal has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision. The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.

Even if the propagation delay is zero, there will still be collisions. If two stations become ready in the middle of a third station's transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision. If they were not so impatient, there would be fewer collisions. Even so, this protocol is far better than pure ALOHA because both stations have the decency to desist from interfering with the third station's frame. Intuitively, this approach will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA.

A second carrier sense protocol is **nonpersistent CSMA**. In this protocol, a conscious attempt is made to be less greedy than in the previous one. Before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Consequently, this algorithm leads to better channel utilization but longer delays than 1-persistent CSMA.

The last protocol is **p-persistent CSMA**. It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p. With a probability $q = 1 - p$, it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q. This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, the unlucky station acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm. Figure 4-4 shows the computed throughput versus offered traffic for all three protocols, as well as for pure and slotted ALOHA.

CSMA with Collision Detection

Persistent and nonpersistent CSMA protocols are clearly an improvement over ALOHA because they ensure that no station begins to transmit when it senses the channel busy. Another improvement is for stations to abort their trans-

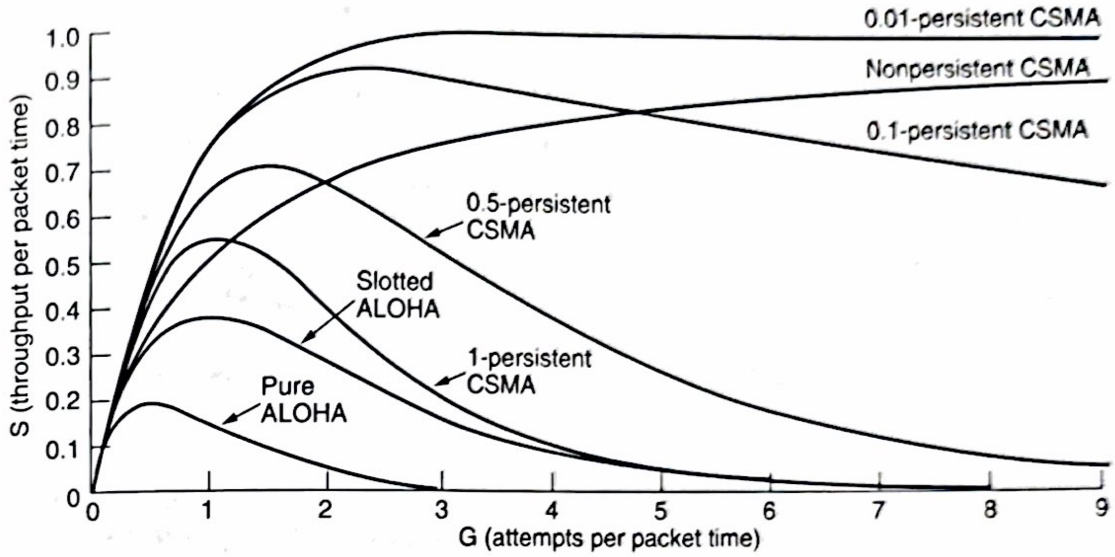

Figure 4-4. Comparison of the channel utilization versus load for various random access protocols.

missions as soon as they detect a collision. In other words, if two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately. Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected. Quickly terminating damaged frames saves time and bandwidth. This protocol, known as **CSMA/CD (CSMA with Collision Detection)** is widely used on LANs in the MAC sublayer. In particular, it is the basis of the popular Ethernet LAN, so it is worth devoting some time to looking at it in detail.

CSMA/CD, as well as many other LAN protocols, uses the conceptual model of Fig. 4-5. At the point marked t_0, a station has finished transmitting its frame. Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision. Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal.

After a station detects a collision, it aborts its transmission, waits a random period of time, and then tries again, assuming that no other station has started transmitting in the meantime. Therefore, our model for CSMA/CD will consist of alternating contention and transmission periods, with idle periods occurring when all stations are quiet (e.g., for lack of work).

Now let us look closely at the details of the contention algorithm. Suppose that two stations both begin transmitting at exactly time t_0. How long will it take them to realize that there has been a collision? The answer to this question is vital to determining the length of the contention period and hence what the delay and throughput will be. The minimum time to detect the collision is then just the time it takes the signal to propagate from one station to the other.

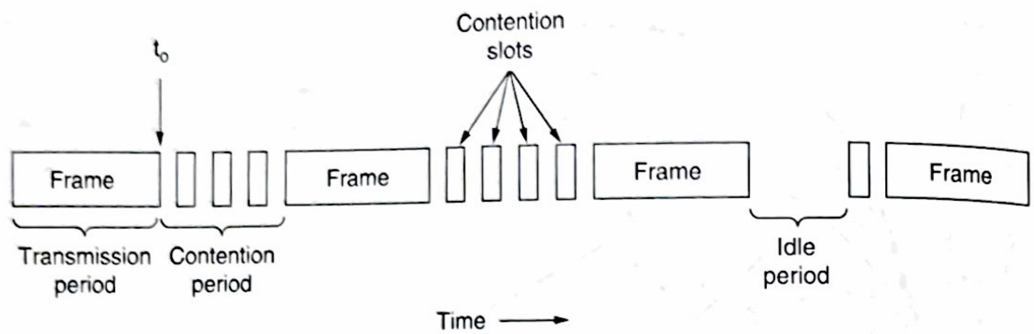

Figure 4-5. CSMA/CD can be in one of three states: contention, transmission, or idle.

Based on this reasoning, you might think that a station not hearing a collision for a time equal to the full cable propagation time after starting its transmission could be sure it had seized the cable. By "seized," we mean that all other stations knew it was transmitting and would not interfere. This conclusion is wrong. Consider the following worst-case scenario. Let the time for a signal to propagate between the two farthest stations be τ. At t_0, one station begins transmitting. At $\tau - \varepsilon$, an instant before the signal arrives at the most distant station, that station also begins transmitting. Of course, it detects the collision almost instantly and stops, but the little noise burst caused by the collision does not get back to the original station until time $2\tau - \varepsilon$. In other words, in the worst case a station cannot be sure that it has seized the channel until it has transmitted for 2τ without hearing a collision. For this reason we will model the contention interval as a slotted ALOHA system with slot width 2τ. On a 1-km long coaxial cable, $\tau \approx 5$ μsec. For simplicity we will assume that each slot contains just 1 bit. Once the channel has been seized, a station can transmit at any rate it wants to, of course, not just at 1 bit per 2τ sec.

It is important to realize that collision detection is an *analog* process. The station's hardware must listen to the cable while it is transmitting. If what it reads back is different from what it is putting out, it knows that a collision is occurring. The implication is that the signal encoding must allow collisions to be detected (e.g., a collision of two 0-volt signals may well be impossible to detect). For this reason, special encoding is commonly used.

It is also worth noting that a sending station must continually monitor the channel, listening for noise bursts that might indicate a collision. For this reason, CSMA/CD with a single channel is inherently a half-duplex system. It is impossible for a station to transmit and receive frames at the same time because the receiving logic is in use, looking for collisions during every transmission.

To avoid any misunderstanding, it is worth noting that no MAC-sublayer protocol guarantees reliable delivery. Even in the absence of collisions, the receiver may not have copied the frame correctly for various reasons (e.g., lack of buffer space or a missed interrupt).

4.2.3 Collision-Free Protocols

Although collisions do not occur with CSMA/CD once a station has unambiguously captured the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the cable is long (i.e., large τ) and the frames are short. And CSMA/CD is not universally applicable. In this section, we will examine some protocols that resolve the contention for the channel without any collisions at all, not even during the contention period. Most of these are not currently used in major systems, but in a rapidly changing field, having some protocols with excellent properties available for future systems is often a good thing.

In the protocols to be described, we assume that there are exactly N stations, each with a unique address from 0 to $N-1$ "wired" into it. It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible. The basic question remains: Which station gets the channel after a successful transmission? We continue using the model of Fig. 4-5 with its discrete contention slots.

A Bit-Map Protocol

In our first collision-free protocol, the **basic bit-map method**, each contention period consists of exactly N slots. If station 0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 during slot 1, but only if it has a frame queued. In general, station j may announce that it has a frame to send by inserting a 1 bit into slot j. After all N slots have passed by, each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting in numerical order (see Fig. 4-6).

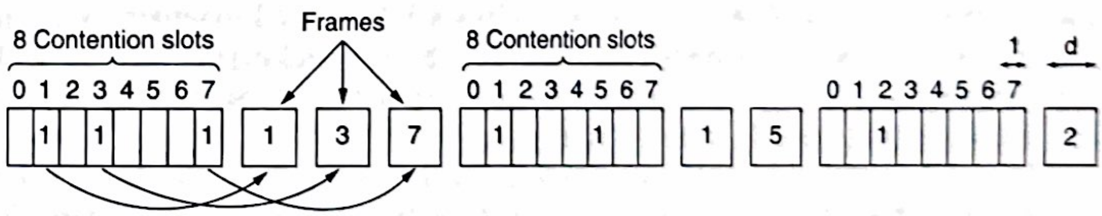

Figure 4-6. The basic bit-map protocol.

Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another N bit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called **reservation protocols**.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of d time units. Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames.

Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the "current" slot will be somewhere in the middle of the bit map. On average, the station will have to wait $N/2$ slots for the current scan to finish and another full N slots for the following scan to run to completion before it may begin transmitting.

The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan ($N/2$ bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan. Since low-numbered stations must wait on average $1.5N$ slots and high-numbered stations must wait on average $0.5N$ slots, the mean for all stations is N slots. The channel efficiency at low load is easy to compute. The overhead per frame is N bits, and the amount of data is d bits, for an efficiency of $d/(N + d)$.

At high load, when all the stations have something to send all the time, the N bit contention period is prorated over N frames, yielding an overhead of only 1 bit per frame, or an efficiency of $d/(d + 1)$. The mean delay for a frame is equal to the sum of the time it queues inside its station, plus an additional $N(d + 1)/2$ once it gets to the head of its internal queue.

Binary Countdown

A problem with the basic bit-map protocol is that the overhead is 1 bit per station, so it does not scale well to networks with thousands of stations. We can do better than that by using binary station addresses. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We will call this protocol **binary countdown**. It was used in Datakit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue.

The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010 because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in Fig. 4-7. It has the property that higher-numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

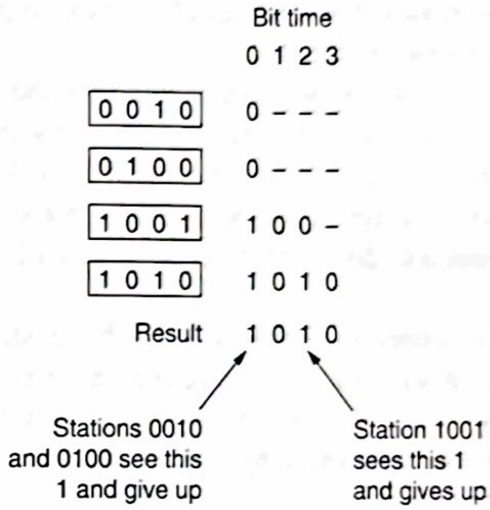

Figure 4-7. The binary countdown protocol. A dash indicates silence.

The channel efficiency of this method is $d/(d + \log_2 N)$. If, however, the frame format has been cleverly chosen so that the sender's address is the first field in the frame, even these $\log_2 N$ bits are not wasted, and the efficiency is 100 percent.

Mok and Ward (1979) have described a variation of binary countdown using a parallel rather than a serial interface. They also suggest using virtual station numbers, with the virtual station numbers from 0 up to and including the successful station being circularly permuted after each transmission, in order to give higher priority to stations that have been silent unusually long. For example, if stations *C, H, D, A, G, B, E, F* have priorities 7, 6, 5, 4, 3, 2, 1, and 0, respectively, then a successful transmission by *D* puts it at the end of the list, giving a priority order of *C, H, A, G, B, E, F, D*. Thus, *C* remains virtual station 7, but *A* moves up from 4 to 5 and *D* drops from 5 to 0. Station *D* will now only be able to acquire the channel if no other station wants it.

Binary countdown is an example of a simple, elegant, and efficient protocol that is waiting to be rediscovered. Hopefully, it will find a new home some day.

4.2.4 Limited-Contention Protocols

We have now considered two basic strategies for channel acquisition in a cable network: contention, as in CSMA, and collision-free methods. Each strategy can be rated as to how well it does with respect to the two important performance measures, delay at low load and channel efficiency at high load. Under conditions of light load, contention (i.e., pure or slotted ALOHA) is preferable due to its low delay. As the load increases, contention becomes increasingly less attractive, because the overhead associated with channel arbitration becomes greater. Just the reverse is true for the collision-free protocols. At low load, they have high delay, but as the load increases, the channel efficiency improves rather than gets worse as it does for contention protocols.

Obviously, it would be nice if we could combine the best properties of the contention and collision-free protocols, arriving at a new protocol that used contention at low load to provide low delay, but used a collision-free technique at high load to provide good channel efficiency. Such protocols, which we will call **limited-contention protocols**, do, in fact, exist, and will conclude our study of carrier sense networks.

Up to now the only contention protocols we have studied have been symmetric, that is, each station attempts to acquire the channel with some probability, p, with all stations using the same p. Interestingly enough, the overall system performance can sometimes be improved by using a protocol that assigns different probabilities to different stations.

Before looking at the asymmetric protocols, let us quickly review the performance of the symmetric case. Suppose that k stations are contending for channel access. Each has a probability p of transmitting during each slot. The probability that some station successfully acquires the channel during a given slot is then $kp(1-p)^{k-1}$. To find the optimal value of p, we differentiate with respect to p, set the result to zero, and solve for p. Doing so, we find that the best value of p is $1/k$. Substituting $p = 1/k$, we get

$$\Pr[\text{success with optimal } p] = \left[\frac{k-1}{k}\right]^{k-1} \qquad (4\text{-}4)$$

This probability is plotted in Fig. 4-8. For small numbers of stations, the chances of success are good, but as soon as the number of stations reaches even five, the probability has dropped close to its asymptotic value of $1/e$.

From Fig. 4-8, it is fairly obvious that the probability of some station acquiring the channel can be increased only by decreasing the amount of competition. The limited-contention protocols do precisely that. They first divide the stations into (not necessarily disjoint) groups. Only the members of group 0 are permitted to compete for slot 0. If one of them succeeds, it acquires the channel and transmits its frame. If the slot lies fallow or if there is a collision, the members of group 1 contend for slot 1, etc. By making an appropriate division of stations into groups, the amount of contention for each slot can be reduced, thus operating each slot near the left end of Fig. 4-8.

The trick is how to assign stations to slots. Before looking at the general case, let us consider some special cases. At one extreme, each group has but one mem-

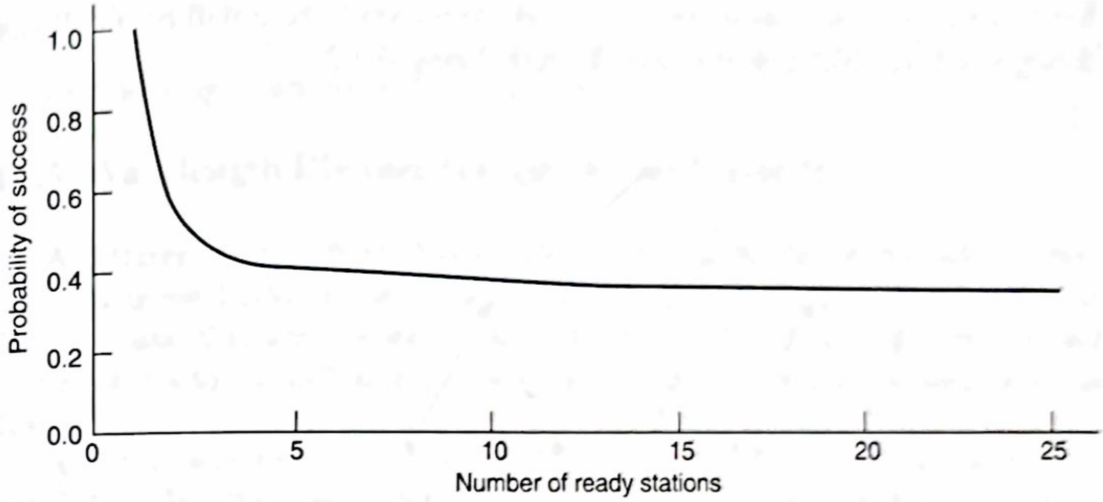

Figure 4-8. Acquisition probability for a symmetric contention channel.

ber. Such an assignment guarantees that there will never be collisions because at most one station is contending for any given slot. We have seen such protocols before (e.g., binary countdown). The next special case is to assign two stations per group. The probability that both will try to transmit during a slot is p^2, which for small p is negligible. As more and more stations are assigned to the same slot, the probability of a collision grows, but the length of the bit-map scan needed to give everyone a chance shrinks. The limiting case is a single group containing all stations (i.e., slotted ALOHA). What we need is a way to assign stations to slots dynamically, with many stations per slot when the load is low and few (or even just one) station per slot when the load is high.

The Adaptive Tree Walk Protocol

One particularly simple way of performing the necessary assignment is to use the algorithm devised by the U.S. Army for testing soldiers for syphilis during World War II (Dorfman, 1943). In short, the Army took a blood sample from N soldiers. A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through $N/2$ and one from the rest. The process was repeated recursively until the infected soldiers were determined.

For the computerized version of this algorithm (Capetanakis, 1979), it is convenient to think of the stations as the leaves of a binary tree, as illustrated in Fig. 4-9. In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4's turn during slot 2.

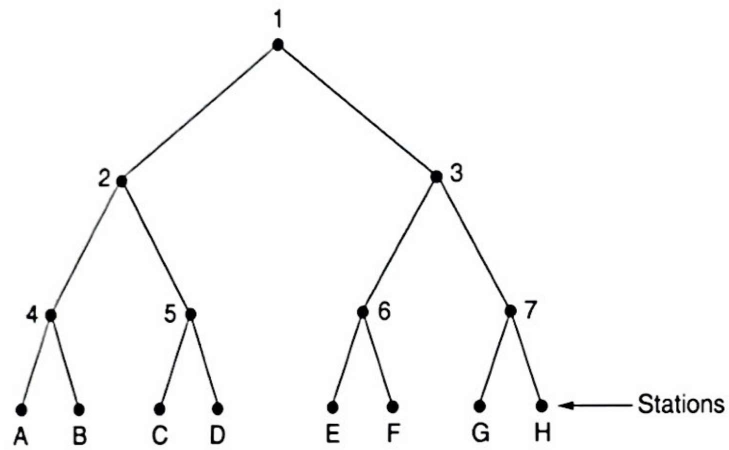

Figure 4-9. The tree for eight stations.

In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node's left and right children. If a bit slot is idle or if only one station transmits in it, the searching of its node can stop because all ready stations have been located. (Were there more than one, there would have been a collision.)

When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1, because that makes sense only in the unlikely event that precisely one station has a frame to send. Similarly, one could argue that nodes 2 and 3 should be skipped as well for the same reason. Put in more general terms, at what level in the tree should the search begin? Clearly, the heavier the load, the farther down the tree the search should begin. We will assume that each station has a good estimate of the number of ready stations, q, for example, from monitoring recent traffic.

To proceed, let us number the levels of the tree from the top, with node 1 in Fig. 4-9 at level 0, nodes 2 and 3 at level 1, etc. Notice that each node at level i has a fraction 2^{-i} of the stations below it. If the q ready stations are uniformly distributed, the expected number of them below a specific node at level i is just $2^{-i}q$. Intuitively, we would expect the optimal level to begin searching the tree as the one at which the mean number of contending stations per slot is 1, that is, the level at which $2^{-i}q = 1$. Solving this equation, we find that $i = \log_2 q$.

Numerous improvements to the basic algorithm have been discovered and are discussed in some detail by Bertsekas and Gallager (1992). For example, consider the case of stations G and H being the only ones wanting to transmit. At node 1 a collision will occur, so 2 will be tried and discovered idle. It is pointless to probe node 3 since it is guaranteed to have a collision (we know that two or more stations under 1 are ready and none of them are under 2, so they must all be under 3). The probe of 3 can be skipped and 6 tried next. When this probe also turns up nothing, 7 can be skipped and node *G* tried next.

4.2.5 Wavelength Division Multiple Access Protocols

A different approach to channel allocation is to divide the channel into subchannels using FDM, TDM, or both, and dynamically allocate them as needed. Schemes like this are commonly used on fiber optic LANs to permit different conversations to use different wavelengths (i.e., frequencies) at the same time. In this section we will examine one such protocol (Humblet et al., 1992).

A simple way to build an all-optical LAN is to use a passive star coupler (see Fig. 2-10). In effect, two fibers from each station are fused to a glass cylinder. One fiber is for output to the cylinder and one is for input from the cylinder. Light output by any station illuminates the cylinder and can be detected by all the other stations. Passive stars can handle hundreds of stations.

To allow multiple transmissions at the same time, the spectrum is divided into channels (wavelength bands), as shown in Fig. 2-31. In this protocol, **WDMA (Wavelength Division Multiple Access)**, each station is assigned two channels. A narrow channel is provided as a control channel to signal the station, and a wide channel is provided so the station can output data frames.

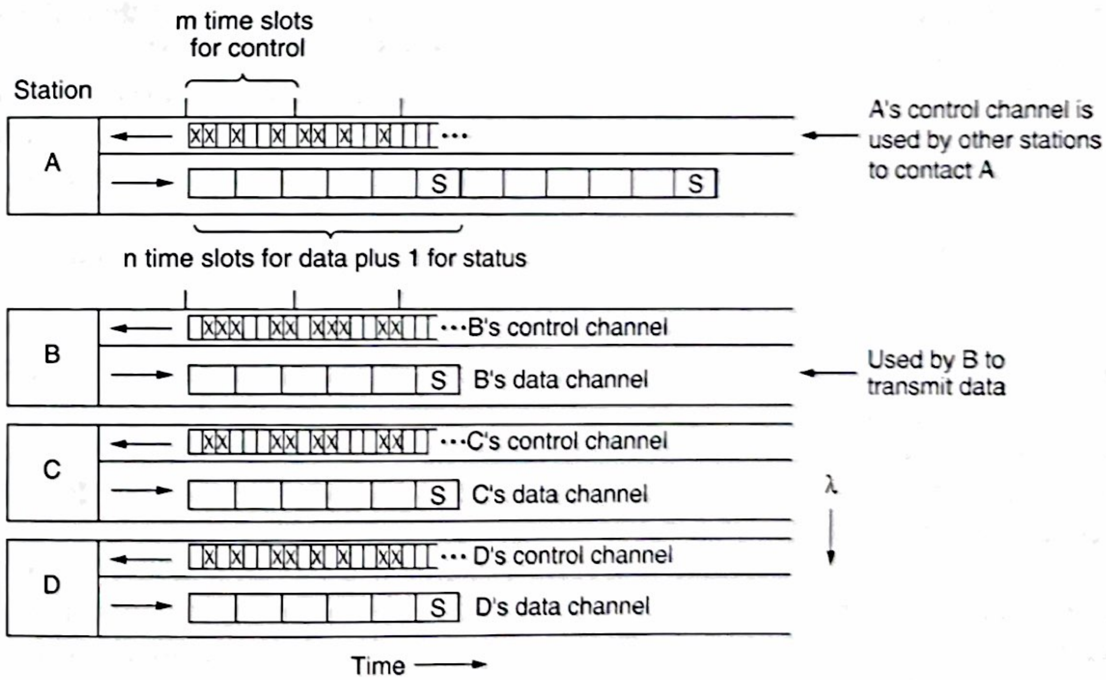

Figure 4-10. Wavelength division multiple access.

Each channel is divided into groups of time slots, as shown in Fig. 4-10. Let us call the number of slots in the control channel *m* and the number of slots in the data channel $n + 1$, where n of these are for data and the last one is used by the station to report on its status (mainly, which slots on both channels are free). On both channels, the sequence of slots repeats endlessly, with slot 0 being marked in a special way so latecomers can detect it. All channels are synchronized by a single global clock.

The protocol supports three traffic classes : (1) constant data rate connection-oriented traffic, such as uncompressed video, (2) variable data rate connection-oriented traffic, such as file transfer, and (3) datagram traffic, such as UDP packets. For the two connection-oriented protocols, the idea is that for *A* to communicate with *B*, it must first insert a CONNECTION REQUEST frame in a free slot on *B*'s control channel. If *B* accepts, communication can take place on *A*'s data channel.

Each station has two transmitters and two receivers, as follows:

1. A fixed-wavelength receiver for listening to its own control channel.
2. A tunable transmitter for sending on other stations' control channels.
3. A fixed-wavelength transmitter for outputting data frames.
4. A tunable receiver for selecting a data transmitter to listen to.

In other words, every station listens to its own control channel for incoming requests but has to tune to the transmitter's wavelength to get the data. Wavelength tuning is done by a Fabry-Perot or Mach-Zehnder interferometer that filters out all wavelengths except the desired wavelength band.

Let us now consider how station *A* sets up a class 2 communication channel with station *B* for, say, file transfer. First, *A* tunes its data receiver to *B*'s data channel and waits for the status slot. This slot tells which control slots are currently assigned and which are free. In Fig. 4-10, for example, we see that of *B*'s eight control slots, 0, 4, and 5 are free. The rest are occupied (indicated by crosses).

A picks one of the free control slots, say, 4, and inserts its CONNECTION REQUEST message there. Since *B* constantly monitors its control channel, it sees the request and grants it by assigning slot 4 to *A*. This assignment is announced in the status slot of *B*'s data channel. When *A* sees the announcement, it knows it has a unidirectional connection. If *A* asked for a two-way connection, *B* now repeats the same algorithm with *A*.

It is possible that at the same time *A* tried to grab *B*'s control slot 4, *C* did the same thing. Neither will get it, and both will notice the failure by monitoring the status slot in *B*'s control channel. They now each wait a random amount of time and try again later.

At this point, each party has a conflict-free way to send short control messages to the other one. To perform the file transfer, *A* now sends *B* a control message saying, for example, "Please watch my next data output slot 3. There is a data frame for you in it." When *B* gets the control message, it tunes its receiver to *A*'s output channel to read the data frame. Depending on the higher-layer protocol, *B* can use the same mechanism to send back an acknowledgement if it wishes.

Note that a problem arises if both *A* and *C* have connections to *B* and each of them suddenly tells *B* to look at slot 3. *B* will pick one of these requests at random, and the other transmission will be lost.

For constant rate traffic, a variation of this protocol is used. When *A* asks for a connection, it simultaneously says something like: Is it all right if I send you a frame in every occurrence of slot 3? If *B* is able to accept (i.e., has no previous commitment for slot 3), a guaranteed bandwidth connection is established. If not, *A* can try again with a different proposal, depending on which output slots it has free.

Class 3 (datagram) traffic uses still another variation. Instead of writing a CONNECTION REQUEST message into the control slot it just found (4), it writes a DATA FOR YOU IN SLOT 3 message. If *B* is free during the next data slot 3, the transmission will succeed. Otherwise, the data frame is lost. In this manner, no connections are ever needed.

Several variants of the protocol are possible. For example, instead of each station having its own control channel, a single control channel can be shared by all stations. Each station is assigned a block of slots in each group, effectively multiplexing multiple virtual channels onto one physical one.

It is also possible to make do with a single tunable transmitter and a single tunable receiver per station by having each station's channel be divided into m control slots followed by $n + 1$ data slots. The disadvantage here is that senders have to wait longer to capture a control slot and consecutive data frames are farther apart because some control information is in the way.

Numerous other WDMA protocols have been proposed and implemented, differing in various details. Some have only one control channel; others have multiple control channels. Some take propagation delay into account; others do not. Some make tuning time an explicit part of the model; others ignore it. The protocols also differ in terms of processing complexity, throughput, and scalability. When a large number of frequencies are being used, the system is sometimes called **DWDM (Dense Wavelength Division Multiplexing)**. For more information see (Bogineni et al., 1993; Chen, 1994; Goralski, 2001; Kartalopoulos, 1999; and Levine and Akyildiz, 1995).

4.2.6 Wireless LAN Protocols

As the number of mobile computing and communication devices grows, so does the demand to connect them to the outside world. Even the very first mobile telephones had the ability to connect to other telephones. The first portable computers did not have this capability, but soon afterward, modems became commonplace on notebook computers. To go on-line, these computers had to be plugged into a telephone wall socket. Requiring a wired connection to the fixed network meant that the computers were portable, but not mobile.

To achieve true mobility, notebook computers need to use radio (or infrared) signals for communication. In this manner, dedicated users can read and send e-mail while hiking or boating. A system of notebook computers that communicate by radio can be regarded as a wireless LAN, as we discussed in Sec. 1.5.4. These LANs have somewhat different properties than conventional LANs and require special MAC sublayer protocols. In this section we will examine some of these protocols. More information about wireless LANs can be found in (Geier, 2002; and O'Hara and Petrick, 1999).

A common configuration for a wireless LAN is an office building with base stations (also called access points) strategically placed around the building. All the base stations are wired together using copper or fiber. If the transmission power of the base stations and notebooks is adjusted to have a range of 3 or 4 meters, then each room becomes a single cell and the entire building becomes a large cellular system, as in the traditional cellular telephony systems we studied in Chap. 2. Unlike cellular telephone systems, each cell has only one channel, covering the entire available bandwidth and covering all the stations in its cell. Typically, its bandwidth is 11 to 54 Mbps.

In our discussions below, we will make the simplifying assumption that all radio transmitters have some fixed range. When a receiver is within range of two active transmitters, the resulting signal will generally be garbled and useless, in other words, we will not consider CDMA-type systems further in this discussion. It is important to realize that in some wireless LANs, not all stations are within range of one another, which leads to a variety of complications. Furthermore, for indoor wireless LANs, the presence of walls between stations can have a major impact on the effective range of each station.

A naive approach to using a wireless LAN might be to try CSMA: just listen for other transmissions and only transmit if no one else is doing so. The trouble is, this protocol is not really appropriate because what matters is interference at the receiver, not at the sender. To see the nature of the problem, consider Fig. 4-11, where four wireless stations are illustrated. For our purposes, it does not matter which are base stations and which are notebooks. The radio range is such that *A* and *B* are within each other's range and can potentially interfere with one another. *C* can also potentially interfere with both *B* and *D*, but not with *A*.

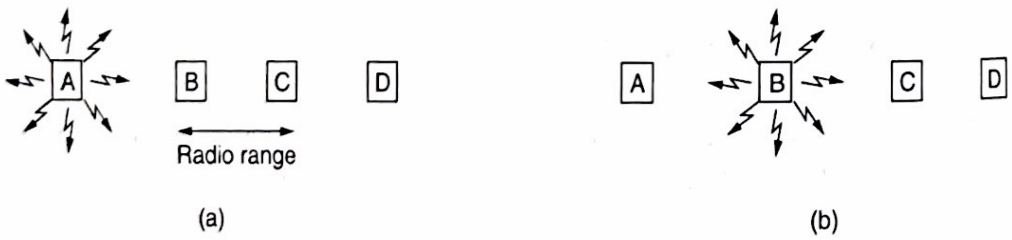

Figure 4-11. A wireless LAN. (a) *A* transmitting. (b) *B* transmitting.

First consider what happens when *A* is transmitting to *B*, as depicted in Fig. 4-11(a). If *C* senses the medium, it will not hear *A* because *A* is out of range, and thus falsely conclude that it can transmit to *B*. If *C* does start transmitting, it will interfere at *B*, wiping out the frame from *A*. The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the **hidden station problem**.

Now let us consider the reverse situation: *B* transmitting to *A*, as shown in Fig. 4-11(b). If *C* senses the medium, it will hear an ongoing transmission and falsely conclude that it may not send to *D*, when in fact such a transmission would cause bad reception only in the zone between *B* and *C*, where neither of the intended receivers is located. This is called the **exposed station problem**.

The problem is that before starting a transmission, a station really wants to know whether there is activity around the receiver. CSMA merely tells it whether there is activity around the station sensing the carrier. With a wire, all signals propagate to all stations so only one transmission can take place at once anywhere in the system. In a system based on short-range radio waves, multiple transmissions can occur simultaneously if they all have different destinations and these destinations are out of range of one another.

Another way to think about this problem is to imagine an office building in which every employee has a wireless notebook computer. Suppose that Linda wants to send a message to Milton. Linda's computer senses the local environment and, detecting no activity, starts sending. However, there may still be a collision in Milton's office because a third party may currently be sending to him from a location so far from Linda that her computer could not detect it.

MACA and MACAW

An early protocol designed for wireless LANs is **MACA (Multiple Access with Collision Avoidance)** (Karn, 1990). The basic idea behind it is for the sender to stimulate the receiver into outputting a short frame, so stations nearby can detect this transmission and avoid transmitting for the duration of the upcoming (large) data frame. MACA is illustrated in Fig. 4-12.

Let us now consider how *A* sends a frame to *B*. *A* starts by sending an **RTS (Request To Send)** frame to *B*, as shown in Fig. 4-12(a). This short frame (30 bytes) contains the length of the data frame that will eventually follow. Then *B* replies with a **CTS (Clear to Send)** frame, as shown in Fig. 4-12(b). The CTS frame contains the data length (copied from the RTS frame). Upon receipt of the CTS frame, *A* begins transmission.

Now let us see how stations overhearing either of these frames react. Any station hearing the RTS is clearly close to *A* and must remain silent long enough for the CTS to be transmitted back to *A* without conflict. Any station hearing the CTS is clearly close to *B* and must remain silent during the upcoming data transmission, whose length it can tell by examining the CTS frame.

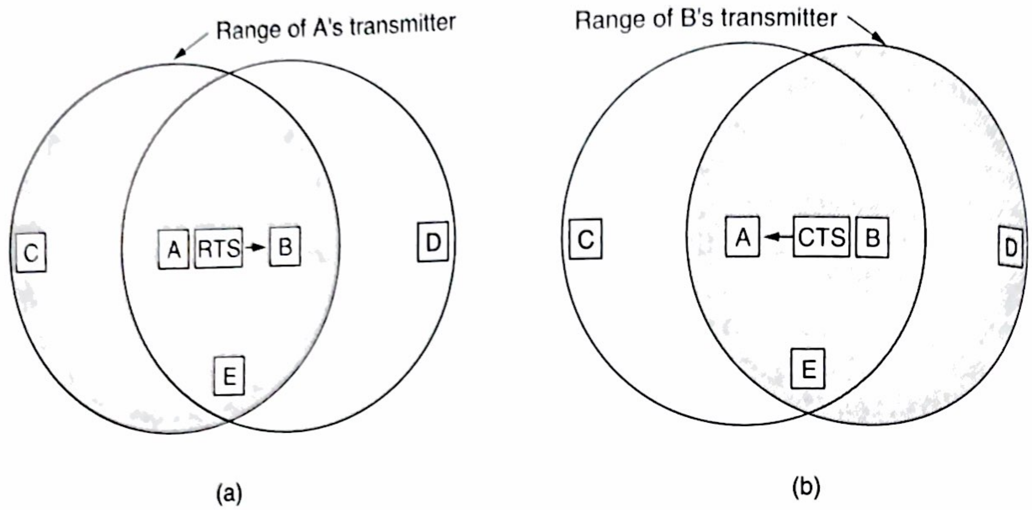

Figure 4-12. The MACA protocol. (a) *A* sending an RTS to *B*. (b) *B* responding with a CTS to *A*.

In Fig. 4-12, *C* is within range of *A* but not within range of *B*. Therefore, it hears the RTS from *A* but not the CTS from *B*. As long as it does not interfere with the CTS, it is free to transmit while the data frame is being sent. In contrast, *D* is within range of *B* but not *A*. It does not hear the RTS but does hear the CTS. Hearing the CTS tips it off that it is close to a station that is about to receive a frame, so it defers sending anything until that frame is expected to be finished. Station *E* hears both control messages and, like *D*, must be silent until the data frame is complete.

Despite these precautions, collisions can still occur. For example, *B* and *C* could both send RTS frames to *A* at the same time. These will collide and be lost. In the event of a collision, an unsuccessful transmitter (i.e., one that does not hear a CTS within the expected time interval) waits a random amount of time and tries again later. The algorithm used is binary exponential backoff, which we will study when we come to Ethernet.

Based on simulation studies of MACA, Bharghavan et al. (1994) fine tuned MACA to improve its performance and renamed their new protocol **MACAW** (**MACA for Wireless**). To start with, they noticed that without data link layer acknowledgements, lost frames were not retransmitted until the transport layer noticed their absence, much later. They solved this problem by introducing an ACK frame after each successful data frame. They also observed that CSMA has some use, namely, to keep a station from transmitting an RTS at the same time another nearby station is also doing so to the same destination, so carrier sensing was added. In addition, they decided to run the backoff algorithm separately for each data stream (source-destination pair), rather than for each station. This change improves the fairness of the protocol. Finally, they added a mechanism for stations to exchange information about congestion and a way to make the backoff algorithm react less violently to temporary problems, to improve system performance.

4.3 ETHERNET

We have now finished our general discussion of channel allocation protocols in the abstract, so it is time to see how these principles apply to real systems, in particular, LANs. As discussed in Sec. 1.5.3, the IEEE has standardized a number of local area networks and metropolitan area networks under the name of IEEE 802. A few have survived but many have not, as we saw in Fig. 1-38. Some people who believe in reincarnation think that Charles Darwin came back as a member of the IEEE Standards Association to weed out the unfit. The most important of the survivors are 802.3 (Ethernet) and 802.11 (wireless LAN). With 802.15 (Bluetooth) and 802.16 (wireless MAN), it is too early to tell. Please consult the 5th edition of this book to find out. Both 802.3 and 802.11 have different physical layers and different MAC sublayers but converge on the same logical link control sublayer (defined in 802.2), so they have the same interface to the network layer.

We introduced Ethernet in Sec. 1.5.3 and will not repeat that material here. Instead we will focus on the technical details of Ethernet, the protocols, and recent developments in high-speed (gigabit) Ethernet. Since Ethernet and IEEE 802.3 are identical except for two minor differences that we will discuss shortly, many people use the terms "Ethernet" and "IEEE 802.3" interchangeably, and we will do so, too. For more information about Ethernet, see (Breyer and Riley, 1999 ; Seifert, 1998; and Spurgeon, 2000).

4.3.1 Ethernet Cabling

Since the name "Ethernet" refers to the cable (the ether), let us start our discussion there. Four types of cabling are commonly used, as shown in Fig. 4-13.

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

Figure 4-13. The most common kinds of Ethernet cabling.

Historically, **10Base5** cabling, popularly called **thick Ethernet,** came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. (The 802.3 standard does not actually *require* the cable to be yellow, but it does *suggest* it.) Connections to it are generally made using **vampire taps,** in which a pin is *very* carefully forced halfway into the coaxial cable's core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. The first number is the speed in Mbps.