Computer network (CMPC-208)

Multiple access techniques, WAN technology and protocols, circuit switching and packet switching.

Packet Switching

In a network application, end systems exchange **messages** with each other. Messages can contain anything the application designer wants. Messages may perform a control function (for example, the "Hi" messages in our handshaking example in Figure 1.2) or can contain data, such as an email message, a JPEG image, or an MP3 audio file. To send a message from a source end system to a destination end system, the source breaks long messages into smaller chunks of data known as **packets**. Between source and destination, each packet travels through communication links and **packet switches** (for which there are two predominant types, **routers** and **link layer switches**). Packets are transmitted over each communication link at a rate equal to the *full* transmission rate of the link. So, if a source end system or a packet switch is sending a packet of L bits over a link with transmission rate R bits/sec, then the time to transmit the packet is L/R seconds.

Store-and-Forward Transmission

Most packet switches use **store-and-forward transmission** at the inputs to the links. Store-and-forward transmission means that the packet switch must receive the entire packet before it can begin to transmit the first bit of the packet onto the outbound link. To explore store-and-forward transmission in more detail, consider a simple network consisting of two end systems connected by a single router, as shown in Figure 1.11. A router will typically have many incident links, since its job is to switch an incoming packet onto an outgoing link; in this simple example, the router has the rather simple task of transferring a packet from one (input) link to the only other attached link. In this example, the source has three packets, each consisting of L bits, to send to the destination. At the snapshot of time shown in Figure 1.11, the source has transmitted some of packet 1, and the front of packet 1 has already arrived at the router. Because the router employs store-and-forwarding, at this instant of time, the router cannot transmit the bits it has received; instead it

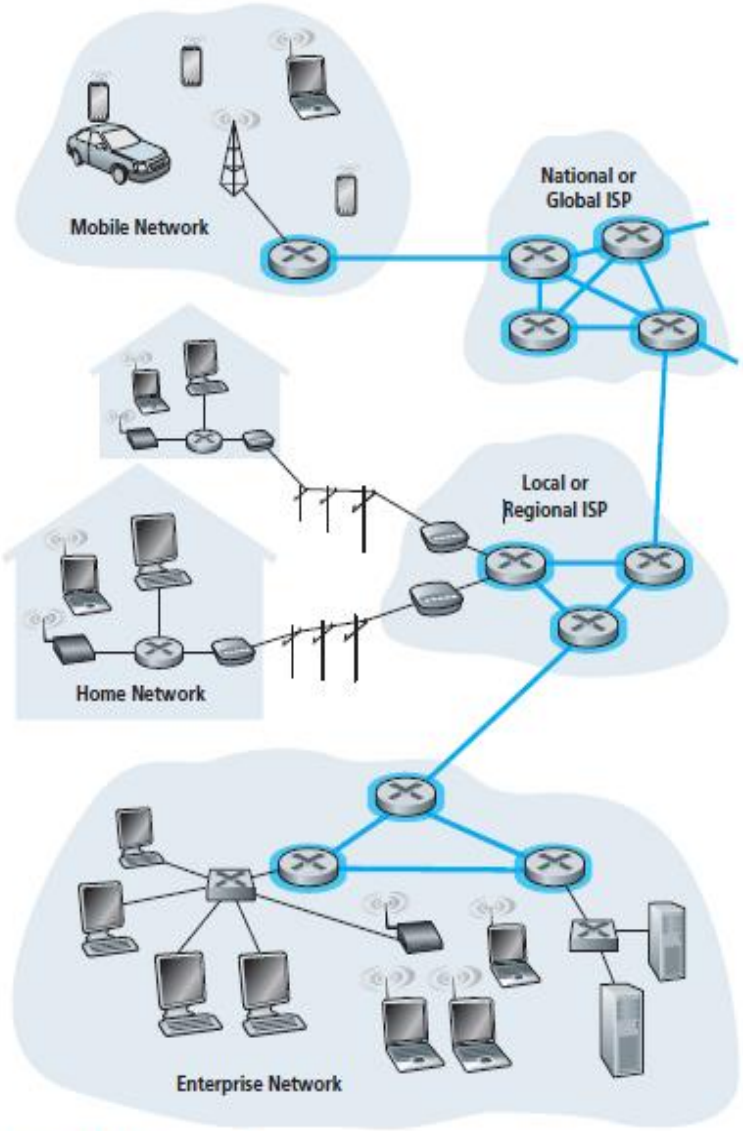

Figure 1.10 ♦ The network core

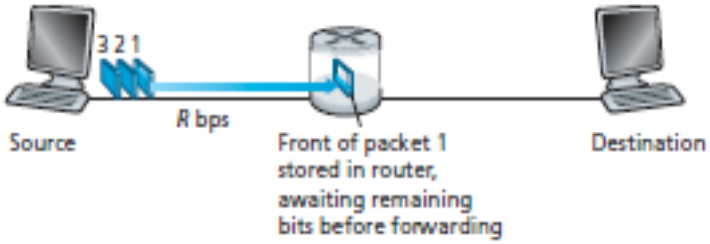

Figure 1.11 ♦ Store-and-forward packet switching

Queuing Delays and Packet Loss

Each packet switch has multiple links attached to it. For each attached link, the packet switch has an **output buffer** (also called an **output queue**), which stores packets that the router is about to send into that link. The output buffers play a key role in packet switching. If an arriving packet needs to be transmitted onto a link but finds the link busy with the transmission of another packet, the arriving packet must wait in the output buffer. Thus, in addition to the store-and-forward delays, packets suffer output buffer **queuing delays**. These delays are variable and depend on the level of congestion in the network. Since the amount of buffer space is finite, an arriving packet may find that the buffer is completely full with other packets waiting for transmission. In this case, **packet loss** will occur—either the arriving packet or one of the already-queued packets will be dropped.
Figure 1.12 illustrates a simple packet-switched network. As in Figure 1.11, packets are represented by three-dimensional slabs. The width of a slab represents the number of bits in the packet. In this figure, all packets have the same width and hence the same length. Suppose Hosts A and B are sending packets to Host E. Hosts A and B first send their packets along 10 Mbps Ethernet links to the first router. The router then directs these packets to the 1.5 Mbps link. If, during a short interval of time, the arrival rate of packets to the router (when converted to bits per second) exceeds 1.5 Mbps, congestion will occur at the router as packets queue in the link's output buffer before being transmitted onto the link. For example, if Host A and B each send a burst of five packets back-to-back at the same time, then most of these packets will spend some time waiting in the queue. The situation is, in fact, entirely analogous to many common-day situations—for example, when we wait in line for a bank teller or wait in front of a tollbooth. We'll examine this queuing delay in more detail in Section 1.4.

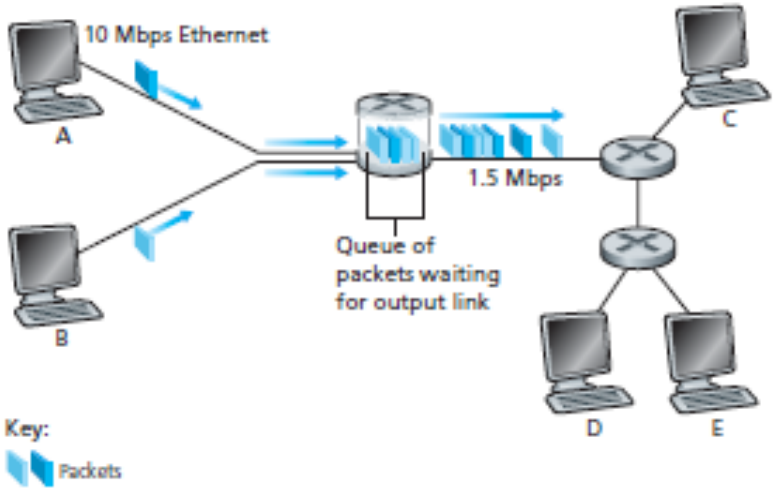

Figure 1.12 ♦ Packet switching

Forwarding Tables and Routing Protocols

Earlier, we said that a router takes a packet arriving on one of its attached communication links and forwards that packet onto another one of its attached communication
links. But how does the router determine which link it should forward the packet onto? Packet forwarding is actually done in different ways in different types of computer networks. Here, we briefly describe how it is done in the Internet.

In the Internet, every end system has an address called an IP address. When a source end system wants to send a packet to a destination end system, the source includes the destination's IP address in the packet's header. As with postal addresses, this address has a hierarchical structure. When a packet arrives at a router in the network, the router examines a portion of the packet's destination address and forwards the packet to an adjacent router. More specifically, each router has a **forwarding table** that maps destination addresses (or portions of the destination addresses) to that router's outbound links. When a packet arrives at a router, the router examines the address and searches its forwarding table, using this destination address, to find the appropriate outbound link. The router then directs the packet to this outbound link.

The end-to-end routing process is analogous to a car driver who does not use maps but instead prefers to ask for directions. For example, suppose Joe is driving from Philadelphia to 156 Lakeside Drive in Orlando, Florida. Joe first drives to his neighborhood gas station and asks how to get to 156 Lakeside Drive in Orlando, Florida. The gas station attendant extracts the Florida portion of the address and tells Joe that he needs to get onto the interstate highway I-95 South, which has an

entrance just next to the gas station. He also tells Joe that once he enters Florida, he should ask someone else there. Joe then takes I-95 South until he gets to Jacksonville, Florida, at which point he asks another gas station attendant for directions.
The attendant extracts the Orlando portion of the address and tells Joe that he should continue on I-95 to Daytona Beach and then ask someone else. In Daytona Beach, another gas station attendant also extracts the Orlando portion of the address and tells Joe that he should take I-4 directly to Orlando. Joe takes I-4 and gets off at the Orlando exit. Joe goes to another gas station attendant, and this time the attendant extracts the Lakeside Drive portion of the address and tells Joe the road he must follow to get to Lakeside Drive. Once Joe reaches Lakeside Drive, he asks a kid on a bicycle how to get to his destination. The kid extracts the 156 portion of the address and points to the house. Joe finally reaches his ultimate destination.
In the above analogy, the gas station attendants and kids on bicycles are analogous to routers.
We just learned that a router uses a packet's destination address to index a forwarding table and determine the appropriate outbound link. But this statement begs yet another question: How do forwarding tables get set? Are they configured by hand in each and every router, or does the Internet use a more automated procedure?
This issue will be studied in depth in Chapter 4. But to whet your appetite here, we'll note now that the Internet has a number of special **routing protocols** that are used to automatically set the forwarding tables. A routing protocol may, for example, determine the shortest path from each router to each destination and use the shortest path results to configure the forwarding tables in the routers.
How would you actually like to see the end-to-end route that packets take in the Internet? We now invite you to get your hands dirty by interacting with the Traceroute program. Simply visit the site www.traceroute.org, choose a source in a particular country, and trace the route from that source to your computer.

Circuit Switching

There are two fundamental approaches to moving data through a network of links and switches: **circuit switching** and **packet switching**. Having covered packet switched networks in the previous subsection, we now turn our attention to circuit switched networks.
In circuit-switched networks, the resources needed along a path (buffers, link transmission rate) to provide for communication between the end systems are *reserved* for the duration of the communication session between the end systems. In packet-switched networks, these resources are *not* reserved; a session's messages use the resources on demand, and as a consequence, may have to wait (that is, queue) for access to a communication link. As a simple analogy, consider two restaurants, one that requires reservations and another that neither requires reservations nor accepts them. For the restaurant that requires reservations, we have to go through the hassle of calling before we leave home. But when we arrive at the restaurant we can, in principle, immediately be seated and order our meal. For the restaurant that does not require reservations, we don't need to bother to reserve a

table. But when we arrive at the restaurant, we may have to wait for a table before we can be seated.
Traditional telephone networks are examples of circuit-switched networks. Consider what happens when one person wants to send information (voice or facsimile) to another over a telephone network. Before the sender can send the information, the network must establish a connection between the sender and the receiver. This is a *bona fide* connection for which the switches on the path between the sender and receiver maintain connection state for that connection. In the jargon of telephony, this connection is called a **circuit**. When the network establishes the circuit, it also reserves a constant transmission rate in the network's links (representing a fraction of each link's transmission capacity) for the duration of the connection. Since a given transmission rate has been reserved for this sender-to-receiver connection, the sender can transfer the data to the receiver at the *guaranteed* constant rate.

Multiplexing in Circuit-Switched Networks

A circuit in a link is implemented with either **frequency-division multiplexing (FDM)** or **time-division multiplexing (TDM)**. With FDM, the frequency spectrum of a link is divided up among the connections established across the link.

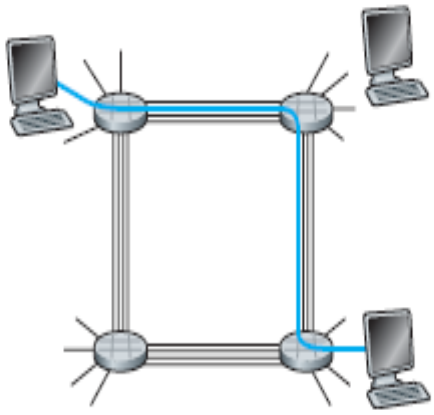

Figure 1.13 ♦ A simple circuit-switched network consisting of four switches and four links

ROUTING

One of the most complex and crucial aspects of packet-switching network design is routing. This section begins with a survey of key characteristics that can be used to classify routing strategies. Then, some specific routing strategies are discussed.

Characteristics

The primary function of a packet-switching network is to accept packets from a source station and deliver them to a destination station. To accomplish this, a path

or route through the network must be determined: generally, more than one route is possible. Thus, a routing function must be performed.

Routing Strategies

A large number of routing strategies have evolved for dealing with the routing requirements of packet-switching networks; many having these strategies are also applied to internetwork routing, which we cover in Part IV. In this section, we survey four key strategies: fixed, flooding, random, and adaptive.

Fixed Routing

For fixed routing, a rote is selected for each source-destination pair of nodes in the network. Either of the least-cost routing algorithms described in Appendix 9A could be used. The routes are fixed, with the exception that they might change if there is movement in the topology of the network. Thus, the link costs used in designing routes cannot be based on any dynamic variable such as traffic. They could, however, be based on expected traffic or capacity.
A central routing matrix is created, to be stored perhaps at a network control center. The matrix shows, for each source-destination pair of nodes, the identity of the next node on the route.

Flooding

Another simple routing technique is flooding. This technique requires no network information whatsoever, and works as follows. A packet is sent by a source node to every one of its neighbors. At each node, an incoming packet is retransmitted on all outgoing links except for the link on which it arrived. For example, if node 1 in Figure 9.5 has a packet to send to node 6, it sends a copy of that packet (with a destination address of 6), to nodes 2,3, and 4. Node 2 will send a copy to nodes 3 and 4. Node 4 will send a copy to nodes 2,3, and 5. And so it goes. Eventually, a number of copies of the packet will arrive at node 6. The packet must have some unique identifier (e.g., source node and sequence number, or virtual-circuit number and sequence number) so that node 6 knows to discard all but the first copy.
Unless something is done to stop the incessant retransmission of packets, the number of packets in circulation just from a single source packet grows without bound; one way to prevent this is for each node to remember the identity of those packets it has already retransmitted. When duplicate copies of the packet arrive, they are discarded. A simpler technique is to include a hop count field with each packet. The count can originally be set to some maximum value, such as the diameter (length of the longest minimum-hop path through the network) of the network. Each time a node passes on a packet, it decrements the count by one. When the count reaches zero, the packet is discarded.

Random Routing

Random routing has the simplicity and robustness of flooding with far less traffic load. With random routing, a node selects only one outgoing path for retransmission of an incoming packet. The outgoing link is chosen at random, excluding the link on which the packet arrived. If all links are equally likely to be chosen, then a node may simply utilize outgoing links in a round-robin fashion.
A refinement of this technique is to assign a probability to each outgoing link and to select the link based on that probability.

Adaptive Routing

In virtually all packet-switching networks, some sort of adaptive routing technique is used. That is, the routing decisions that are made change as conditions on the network change. The principle conditions that influence routing decisions are
Failure. When a node or trunk fails, it can no longer be used as part of a route.
Congestion. When a particular portion of the network is heavily congested, it is desirable to route packets around, rather than through, the area of congestion.
For adaptive routing to be possible, information about the state of the network must be exchanged among the nodes. There is a tradeoff here between the quality of the information and the amount of overhead. The more information that is exchanged, and the more frequently it is exchanged, the better will be the routing decisions that each node makes. On the other hand, this information is itself a load on the network, causing a performance degradation.
There are several drawbacks associated with the use of adaptive routing:
The routing decision is more complex; therefore, the processing burden on network nodes increases.
In most cases, adaptive strategies depend on status information that is collected at one place but used at another; therefore, the traffic burden on the network increases.
An adaptive strategy may react too quickly, causing congestion-producing oscillation; if it reacts too slowly, the strategy will be irrelevant.
Despite these real dangers, adaptive routing strategies are by far the most prevalent, for two reasons:
An adaptive routing strategy can improve performance, as seen by the network user.
An adaptive routing strategy can aid in congestion control, as discussed later.

CONGESTION CONTROL

As with routing, the concept of traffic control in a packet-switching network is complex, and a wide variety of approaches have been proposed. The objective here is to maintain the number of packets within the network below the level at which performance falls off dramatically.
To understand the issue involved in congestion control, we need to look at some results from queuing theory. In essence, a packet-switching network is a network of queues. At each node, there is a queue of packets for each outgoing channel.

If the rate at which packets arrive and queue up exceeds the rate at which packets can be transmitted, the queue size grows without bound and the delay experienced by a packet goes to infinity. Even if the packet arrival rate is less than the packet transmission rate, queue length will grow dramatically as the arrival rate approaches the transmission rate.
As a rule of thumb, when the line for which packets are queuing becomes more than 80% utilized, the queue length grows at an alarming rate.
Consider the queuing situation at a single packet-switching node, such as is illustrated in Figure 9.13. Any given node has a number of transmission links attached to it: one or more to other packet-switching nodes, and zero or more to host systems. On each link, packets arrive and depart. We can consider that there are two buffers at each link, one to accept arriving packets, and one to hold packets that are waiting to depart. In practice, there might be two fixed-size buffers associated with each link, or there might be a pool of memory available for all buffering activities. In the latter case, we can think of each link having two variable-size buffers associated with it, subject to the constraint that the sum of all buffer sizes is a constant.

X .25

Perhaps the best-known and most widely used protocol standard is X.25, which was originally approved in 1976 and subsequently revised in 1980,1984,1988,1992, and 1993. The standard specifies an interface between a host system and a packet switching network. This standard is almost universally used for interfacing to packet-switching networks and is employed for packet switching in ISDN. In this section, a brief overview of the standard is provided.
The standard specifically calls for three layers of functionality.

* Physical layer
* Link layer
*Packet layer

These three layers correspond to the lowest three layers of the OSI model (see Figure 1.10). The physical layer deals with the physical interface between an attached station (computer, terminal) and the link that attaches that station to the packet-switching node. The standard refers to user machines as data terminal equipment (DTE) and to a packet-switching node to which a DTE is attached as data circuit-terminating equipment (DCE). X.25 makes use of the physical-layer specification in a standard known as X.21, but, in many cases, other standards, such as EIA-232, are substituted. The link layer provides for the reliable transfer of data across the physical link by transmitting the data as a sequence of frames. The linklayer standard is referred to as LAPB (Link Access Protocol-Balanced). LAPB is a subset of HDLC, described in Chapter 6. The packet layer provides an external virtual-circuit service, and is described in this section.
Figure 9.17 illustrates the relationship between the levels of X.25. User data are passed down to X.25 level 3, which appends control information as a header,

creating a ***packet.*** This control information is used in the operation of the protocol, as we shall see. The entire X.25 packet is then passed down to the LAPB entity, which appends control information at the front and back of the packet, forming an LAPB *frame.* Again, the control information in the frame is needed for the operation of the LAPB protocol.

Virtual Circuit Service

With the X.25 packet layer, data are transmitted in packets over external virtual circuits. The virtual-circuit service of X.25 provides for two types of virtual circuit: virtual call and permanent virtual circuit. A *virtual call* is a dynamically established virtual circuit using a call setup and call clearing procedure, explained below. A *permanent virtual circuit* is a fixed, network-assigned virtual circuit. Data transfer occurs as with virtual calls, but no call setup or clearing is required.
The left-hand part of the figure shows the packets exchanged between user machine A and the packet-switching node to which it attaches; the right-hand part shows the packets exchanged between user machine B and its node. The routing of packets inside the network is not visible to the user.

Multiplexing

In data transmission, a function that permits two or more data sources to share a common transmission medium such that each data source has its own channel.