COMPUTER NETWORKS

Lecture Notes
UNIT-I

RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS::ONGOLE
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

(Approved by AICTE, Affiliated to JNTU Kakinada)

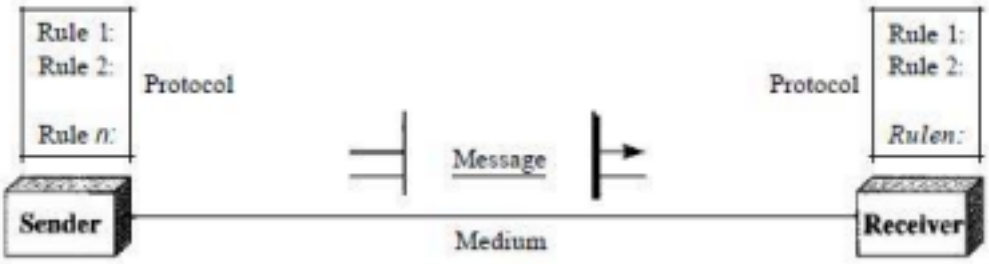
UNIT -I
Introduction to Computer Networks

1.1 Data Communication: When we communicate, we are sharing information. This sharing can be local or remote. Between individuals, local communication usually occurs face to face, while remote communication takes place over distance.

Computer Network: A computer network is a set of computers connected together for the purpose of sharing resources. The most common resource shared today is connection to the Internet. Other shared resources can include a printer or a file server. The Internet itself can be considered a computer network.

1.1.1 Components:

A data communications system has five components.



1. Message. The message is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.

2. Sender. The sender is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.

3. Receiver. The receiver is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.

4. Transmission medium. The transmission medium is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves

5. Protocol. A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating, just as a person speaking French cannot be understood by a person who speaks only Japanese.

1.1.2 Data Representation:

Information today comes in different forms such as text, numbers, images, audio, and video. *Text:*

In data communications, text is represented as a bit pattern, a sequence of bits (Os or Is). Different sets of bit patterns have been designed to represent text symbols. Each set is called a code, and the process of representing symbols is called coding. Today, the prevalent coding system is called Unicode, which uses 32 bits to represent a symbol or character used in any

language in the world. The American Standard Code for Information Interchange (ASCII), developed some decades ago in the United States, now constitutes the first 127 characters in Unicode and is also referred to as Basic Latin.

Numbers:

Numbers are also represented by bit patterns. However, a code such as ASCII is not used to represent numbers; the number is directly converted to a binary number to simplify mathematical operations. Appendix B discusses several different numbering systems. *Images:*

Images are also represented by bit patterns. In its simplest form, an image is composed of a matrix of pixels (picture elements), where each pixel is a small dot. The size of the pixel depends on the *resolution.* For example, an image can be divided into 1000 pixels or 10,000 pixels. In the second case, there is a better representation of the image (better resolution), but more memory is needed to store the image. After an image is divided into pixels, each pixel is assigned a bit pattern. The size and the value of the pattern depend on the image. For an image made of only blackand- white dots (e.g., a chessboard), a I-bit pattern is enough to represent a pixel. If an image is not made of pure white and pure black pixels, you can increase the size of the bit pattern to include gray scale. For example, to show four levels of gray scale, you can use 2-bit patterns. A black pixel can be represented by 00, a dark gray pixel by 01, a light gray pixel by 10, and a white pixel by 11. There are several methods to represent color images. One method is called RGB, so called because each color is made of a combination of three primary colors: *red,* green, and blue. The intensity of each color is measured, and a bit pattern is assigned to it. Another method is called YCM, in which a color is made of a combination of three other primary colors: yellow, cyan, and magenta.

Audio:

Audio refers to the recording or broadcasting of sound or music. Audio is by nature different from text, numbers, or images. It is continuous, not discrete. Even when we use a microphone to change voice or music to an electric signal, we create a continuous signal. In Chapters 4 and 5, we learn how to change sound or music to a digital or an analog signal.

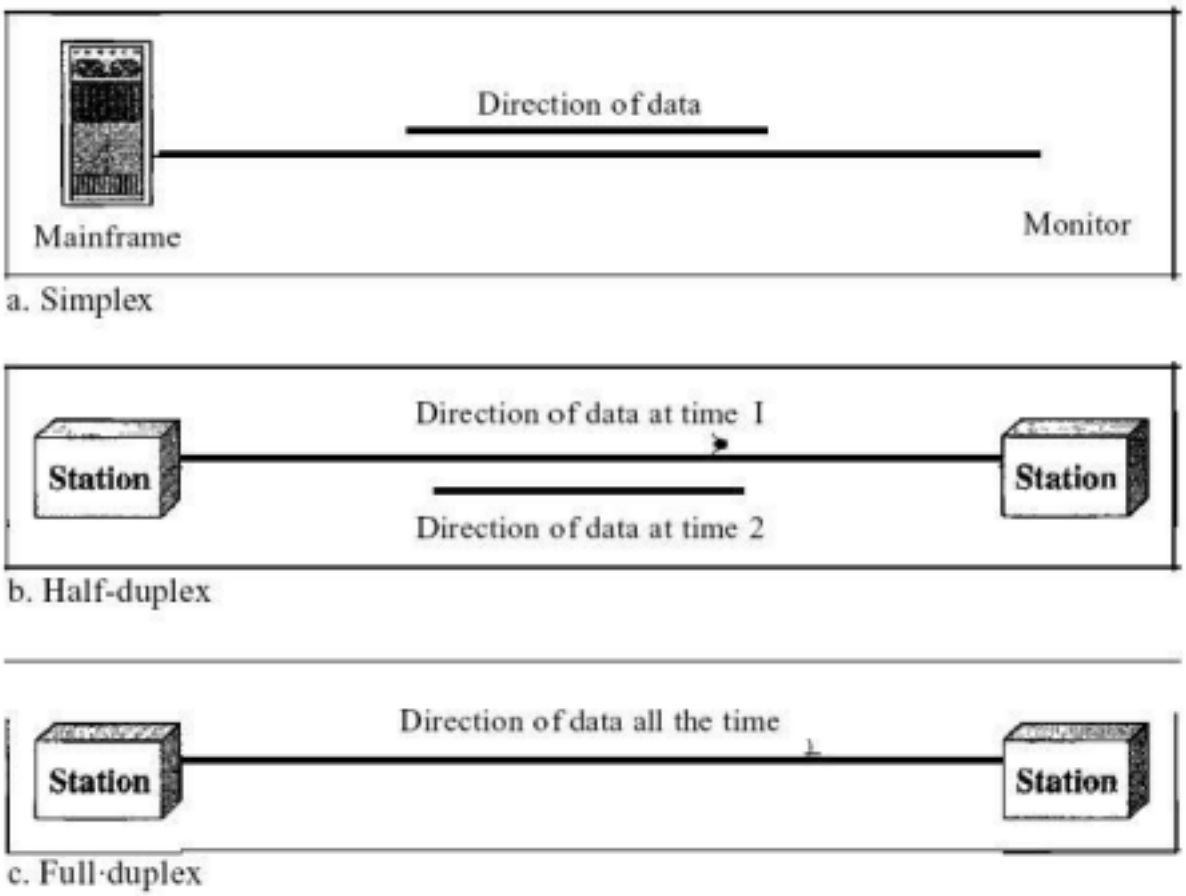
Video:

Video refers to the recording or broadcasting of a picture or movie. Video can either be produced as a continuous entity (e.g., by a TV camera), or it can be a combination of images, each a discrete entity, arranged to convey the idea of motion. Again we can change video to a

digital or an analog signal.

1.1.3 Data Flow

Communication between two devices can be simplex, half-duplex, or full-duplex as shown in Figure



Simplex:

In simplex mode, the communication is unidirectional, as on a one-way street. Only one of the two devices on a link can transmit; the other can only receive (see Figure a). Keyboards and traditional monitors are examples of simplex devices. The keyboard can only introduce input; the monitor can only accept output. The simplex mode can use the entire capacity of the channel to send data in one direction.

Half-Duplex:

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa The half-duplex mode is

like a one-lane road with traffic allowed in both directions.

When cars are traveling in one direction, cars going the other way must wait. In a half-duplex transmission, the entire capacity of a channel is taken over by whichever of the two devices is transmitting at the time. Walkie-talkies and CB (citizens band) radios are both half-duplex systems.

The half-duplex mode is used in cases where there is no need for communication in both directions at the same time; the entire capacity of the channel can be utilized for each direction.

Full-Duplex:

In full-duplex both stations can transmit and receive simultaneously (see Figure c). The full-duplex mode is like a tW<D-way street with traffic flowing in both directions at the same time. In full-duplex mode, si~nals going in one direction share the capacity of the link: with signals going in the other din~c~on. This sharing can occur in two ways: Either the link must contain two physically separate t:nmsmissiIDn paths, one for sending and the other for receiving; or the capacity of the ch:arillilel is divided between signals traveling in both directions. One common example of full-duplex communication is the telephone network. When two people are communicating by a telephone line, both can talk and listen at the same time. The full-duplex mode is used when communication in both directions is required all the time. The capacity of the channel, however, must be divided between the two directions.

1.2 NETWORKS

A network is a set of devices (often referred to as *nodes)* connected by communication links. A node can be a computer, printer, or any other device capable of sending and/or receiving data generated by other nodes on the network.

1.2.1 Distributed Processing

Most networks use distributed processing, in which a task is divided among multiple computers. Instead of one single large machine being responsible for all aspects of a process, separate computers (usually a personal computer or workstation) handle a subset.

1.2.2 Network Criteria

A network must be able to meet a certain number of criteria. The most important of these are performance, reliability, and security.

Performance:

Performance can be measured in many ways, including transit time and response time.Transit time is the amount of time required for a message to travel from one device to another. Response time is the elapsed time between an inquiry and a response. The performance of a network depends on a number of factors, including the number of users, the type of transmission medium, the capabilities of the connected hardware, and the efficiency of the software. Performance is often evaluated by two networking metrics: throughput and delay. We often need more throughput and less delay. However, these two criteria are often contradictory. If we try to send more data to the network, we may increase throughput but we increase the delay because of traffic congestion in the network.

Reliability:

In addition to accuracy of delivery, network reliability is measured by the frequency of failure, the time it takes a link to recover from a failure, and the network's robustness in a catastrophe.

Security:

Network security issues include protecting data from unauthorized access, protecting data from damage and development, and implementing policies and procedures for recovery from breaches and data losses.

1.2.3 Physical Structures:

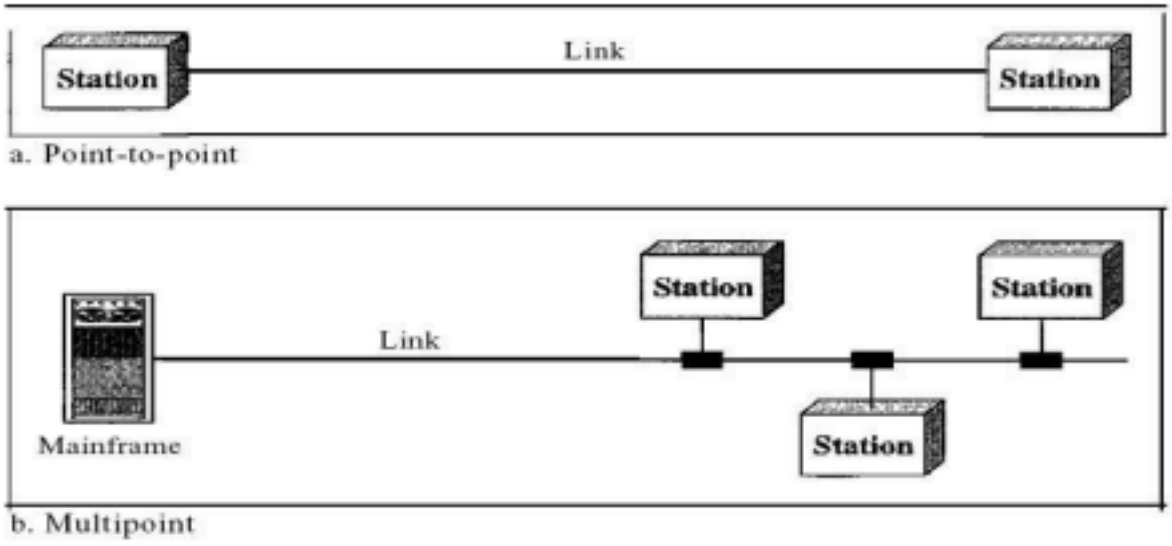
Type of Connection

A network is two or more devices connected through links. A link is a communications pathway that transfers data from one device to another. For visualization purposes, it is simplest to imagine any link as a line drawn between two points. For communication to occur, two devices must be connected in some way to the same link at the same time. There are two possible types of connections: point-to-point and multipoint. Point-to-Point

A point-to-point connection provides a dedicated link between two devices. The entire capacity of the link is reserved for transmission between those two devices. Most point-to-point connections use an actual length of wire or cable to connect the two ends, but other options, such as microwave or satellite links, are also possible. When you change television channels by infrared remote control, you are establishing a point-to-point connection between the remote control and the television's control system.

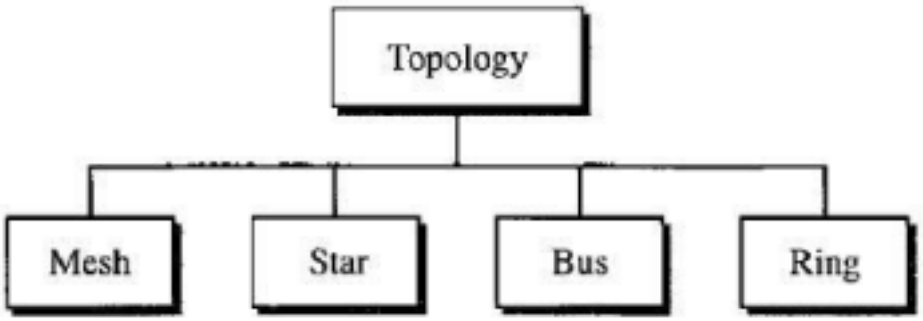
Multipoint

A multipoint (also called multidrop) connection is one in which more than two specific devices share a single link. In a multipoint environment, the capacity of the channel is shared, either spatially or temporally. If several devices can use the link simultaneously, it is a *spatially shared* connection. If users must take turns, it is a *timeshared* connection.



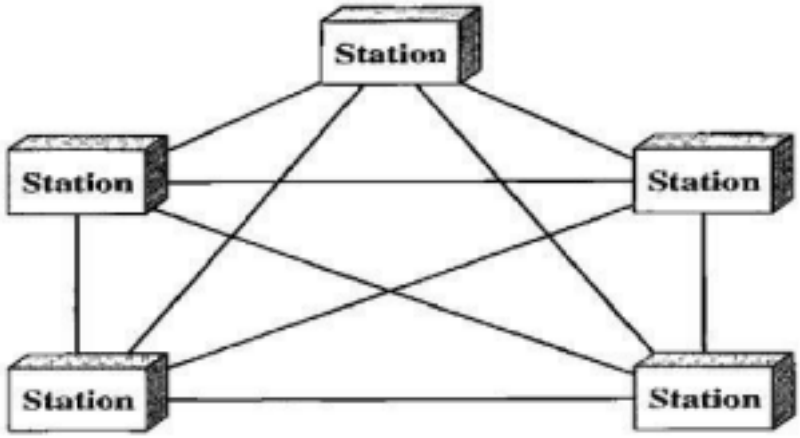
1.2.3.1 Physical Topology

The term *physical topology* refers to the way in which a network is laid out physically. One or more devices connect to a link; two or more links form a topology. The topology of a network is the geometric representation of the relationship of all the links and linking devices (usually called nodes) to one another. There are four basic topologies possible: mesh, star, bus, and ring



Mesh: In a mesh topology, every device has a dedicated point-to-point link to every other device. The term *dedicated* means that the link carries traffic only between the two devices it connects. To find the number of physical links in a fully connected mesh network with *n* nodes, we first consider that each node must be connected to every other node. Node 1 must be connected to *n* - I nodes, node 2 must be connected to *n* – 1 nodes, and finally node *n* must be connected to *n* - 1 nodes. We need *n(n* - 1) physical links. However, if each physical link allows communication in both directions (duplex mode), we can divide the number of links by 2. In other words, we can say that in a mesh topology, we need *n(n* -1) /2 duplex-mode links.

To accommodate that many links, every device on the network must have *n* – 1 input/output *(VO)* ports to be connected to the other *n* - 1 stations.



Advantages:

1. The use of dedicated links guarantees that each connection can carry its own data load, thus eliminating the traffic problems that can occur when links must be shared by multiple devices.
2. A mesh topology is robust. If one link becomes unusable, it does not incapacitate the

entire system. Third, there is the advantage of privacy or security. When every message travels along a dedicated line, only the intended recipient sees it. Physical boundaries prevent other users from gaining access to messages. Finally, point-to-point links make fault identification and fault isolation easy. Traffic can be routed to avoid links with suspected problems. This facility enables the network manager to discover the precise location of the fault and aids in finding its cause and solution.

1. Disadvantage of a mesh are related to the amount of cabling because every device must be connected to every other device, installation and reconnection are difficult. 2. Second, the sheer bulk of the wiring can be greater than the available space (in walls, ceilings, or floors) can accommodate. Finally, the hardware required to connect each link (I/O ports and cable) can be prohibitively expensive.

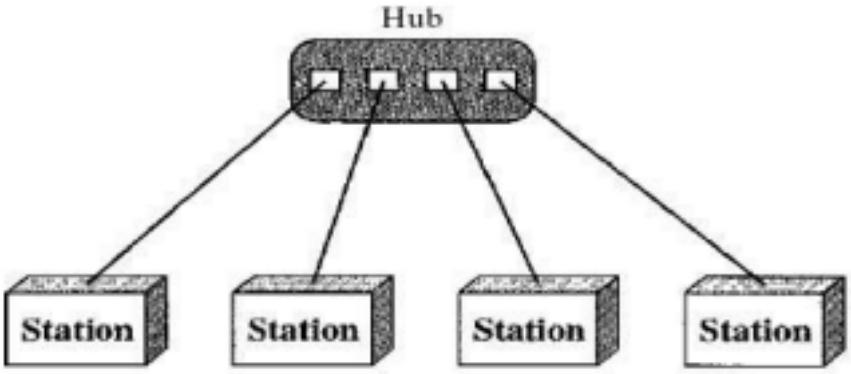
For these reasons a mesh topology is usually implemented in a limited fashion, for example, as a backbone connecting the main computers of a hybrid network that can include several other topologies.

Star Topology:

In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub. The devices are not directly linked to one another. Unlike a mesh topology, a star topology does not allow direct traffic between devices. The controller acts as an exchange: If one device wants to send data to another, it sends the data to the controller, which then relays the data to the other connected device .

A star topology is less expensive than a mesh topology. In a star, each device needs only one link and one I/O port to connect it to any number of others. This factor also makes it easy to install and reconfigure. Far less cabling needs to be housed, and additions, moves, and deletions involve only one connection: between that device and the hub.

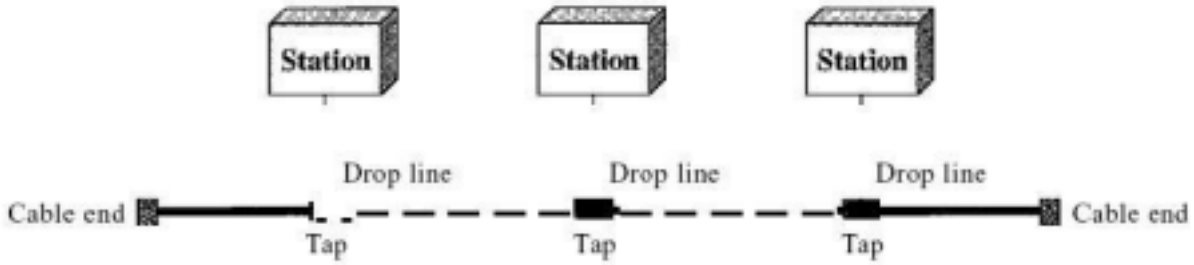
Other advantages include robustness. If one link fails, only that link is affected. All other links remain active. This factor also lends itself to easy fault identification and fault isolation. As long as the hub is working, it can be used to monitor link problems and bypass defective links.



One big disadvantage of a star topology is the dependency of the whole topology on one single point, the hub. If the hub goes down, the whole system is dead. Although a star requires far less cable than a mesh, each node must be linked to a central hub. For this reason, often more cabling is required in a star than in some other topologies (such as ring or bus).

Bus Topology:

The preceding examples all describe point-to-point connections. A **bus topology,** on the other hand, is multipoint. One long cable acts as a **backbone** to link all the devices in a network



Nodes are connected to the bus cable by drop lines and taps. A drop line is a connection running between the device and the main cable. A tap is a connector that either splices into the main cable or punctures the sheathing of a cable to create a contact with the metallic core. As a signal travels along the backbone, some of its energy is transformed into heat. Therefore, it becomes weaker and weaker as it travels farther and farther. For this reason there is a limit on the number of taps a bus can support and on the distance between those taps.

Advantages of a bus topology include ease of installation. Backbone cable can be laid along the most efficient path, then connected to the nodes by drop lines of various lengths. In this way, a bus uses less cabling than mesh or star topologies. In a star, for example, four network devices in

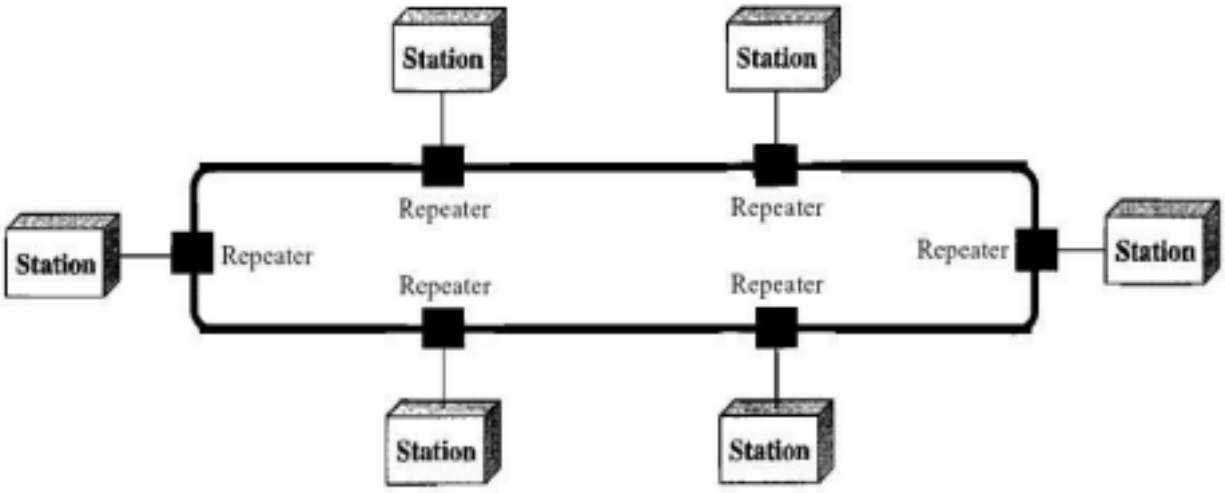
the same room require four lengths of cable reaching all the way to the hub. In a bus, this redundancy is eliminated. Only the backbone cable stretches through the entire facility. Each drop line has to reach only as far as the nearest point on the backbone.

Disadvantages include difficult reconnection and fault isolation. A bus is usually designed to be optimally efficient at installation. It can therefore be difficult to add new devices. Signal reflection at the taps can cause degradation in quality. This degradation can be controlled by limiting the number and spacing of devices connected to a given length of cable. Adding new devices may therefore require modification or replacement of the backbone.

In addition, a fault or break in the bus cable stops all transmission, even between devices on the same side of the problem. The damaged area reflects signals back in the direction of origin, creating noise in both directions.

Bus topology was the one of the first topologies used in the design of early local area networks. Ethernet LANs can use a bus topology, but they are less popular.

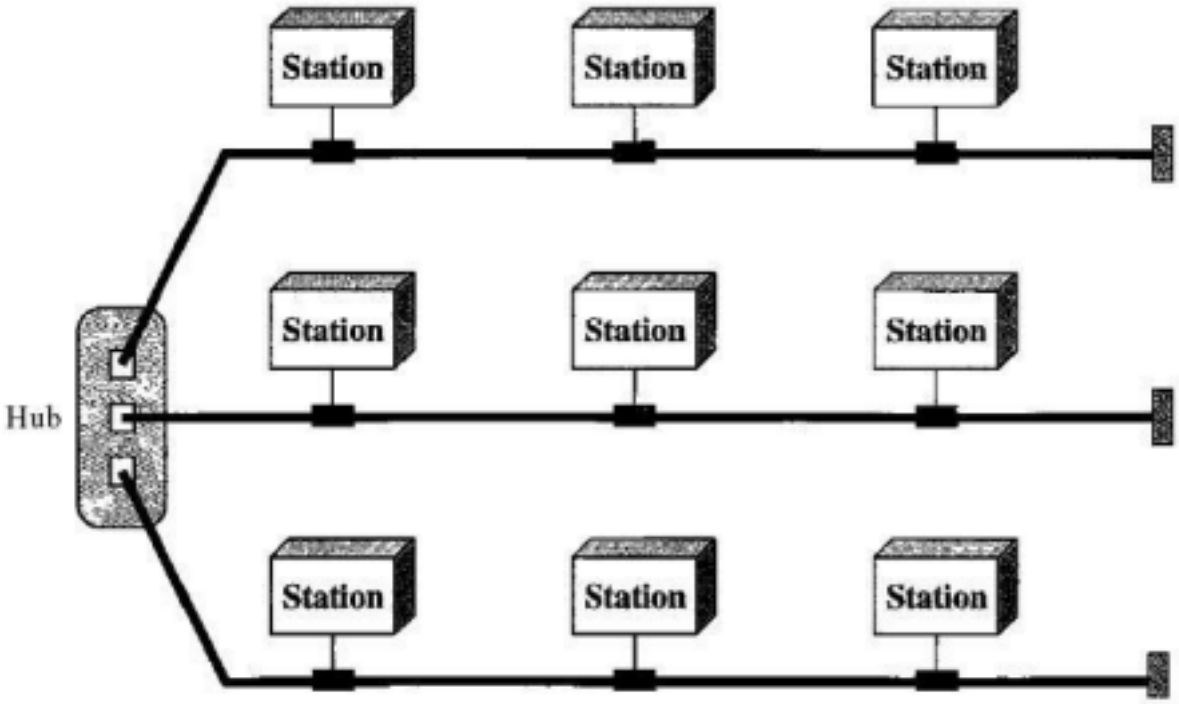
Ring Topology In a ring topology, each device has a dedicated point-to-point connection with only the two devices on either side of it. A signal is passed along the ring in one direction, from device to device, until it reaches its destination. Each device in the ring incorporates a repeater. When a device receives a signal intended for another device, its repeater regenerates the bits and passes them along



A ring is relatively easy to install and reconfigure. Each device is linked to only its immediate neighbors (either physically or logically). To add or delete a device requires changing only two connections. The only constraints are media and traffic considerations (maximum ring length and number of devices). In addition, fault isolation is simplified. Generally in a ring, a signal is

circulating at all times. If one device does not receive a signal within a specified period, it can issue an alarm. The alarm alerts the network operator to the problem and its location.

However, unidirectional traffic can be a disadvantage. In a simple ring, a break in the ring (such as a disabled station) can disable the entire network. This weakness can be solved by using a dual ring or a switch capable of closing off the break. Ring topology was prevalent when IBM introduced its local-area network Token Ring. Today, the need for higher-speed LANs has made this topology less popular. Hybrid Topology A network can be hybrid. For example, we can have a main star topology with each branch connecting several stations in a bus topology as shown in Figure



1.2.4 Categories of Networks

Local Area Networks:

Local area networks, generally called LANs, are privately-owned networks within a single building or campus of up to a few kilometres in size. They are widely used to connect personal computers and workstations in company offices and factories to share resources (e.g., printers)

and exchange information. LANs are distinguished from other kinds of networks by three characteristics:

(1) Their size,

(2) Their transmission technology, and

(3) Their topology.

LANs are restricted in size, which means that the worst-case transmission time is bounded and known in advance. Knowing this bound makes it possible to use certain kinds of designs that would not otherwise be possible. It also simplifies network management. LANs may use a transmission technology consisting of a cable to which all the machines are attached, like the telephone company party lines once used in rural areas. Traditional LANs run at speeds of 10 Mbps to 100 Mbps, have low delay (microseconds or nanoseconds), and make very few errors. Newer LANs operate at up to 10 Gbps Various topologies are possible for broadcast LANs. Figure1 shows two of them. In a bus (i.e., a linear cable) network, at any instant at most one machine is the master and is allowed to transmit. All other machines are required to refrain from sending. An arbitration mechanism is needed to resolve conflicts when two or more machines want to transmit simultaneously. The arbitration mechanism may be centralized or distributed. IEEE 802.3, popularly called Ethernet, for example, is a bus-based broadcast network with decentralized control, usually operating at 10 Mbps to 10 Gbps. Computers on an Ethernet can transmit whenever they want to; if two or more packets collide, each computer just waits a random time and tries again later.

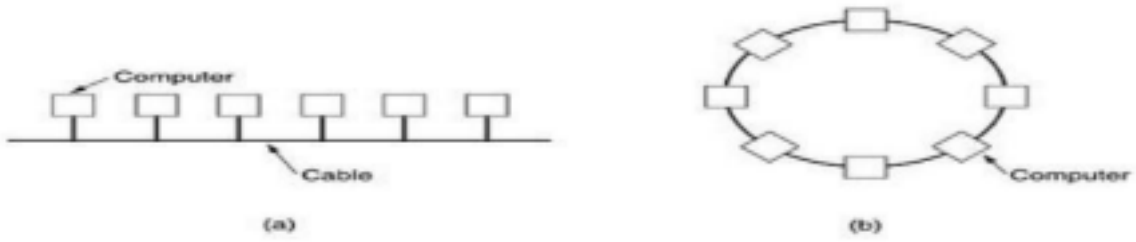


Fig.1: Two broadcast networks . (a) Bus. (b) Ring.

A second type of broadcast system is the ring. In a ring, each bit propagates around on its own, not waiting for the rest of the packet to which it belongs. Typically, each bit circumnavigates the entire ring in the time it takes to transmit a few bits, often before the complete packet has even been transmitted. As with all other broadcast systems, some rule is needed for arbitrating simultaneous accesses to the ring. Various methods, such as having the machines take turns, are in use. IEEE 802.5 (the IBM token ring), is a ring-based LAN operating at 4 and 16 Mbps. FDDI

is another example of a ring network.

Metropolitan Area Network (MAN):

Metropolitan Area Network:

A metropolitan area network, or MAN, covers a city. The best-known example of a MAN is the cable television network available in many cities. This system grew from earlier community antenna systems used in areas with poor over-the-air television reception. In these early systems, a large antenna was placed on top of a nearby hill and signal was then piped to the subscribers' houses. At first, these were locally-designed, ad hoc systems. Then companies began jumping into the business, getting contracts from city governments to wire up an entire city. The next step was television programming and even entire channels designed for cable only. Often these channels were highly specialized, such as all news, all sports, all cooking, all gardening, and so on. But from their inception until the late 1990s, they were intended for television reception only. To a first approximation, a MAN might look something like the system shown in Fig. In this figure both television signals and Internet are fed into the centralized head end for subsequent distribution to people's homes. Cable television is not the only MAN. Recent developments in high-speed wireless Internet access resulted in another MAN, which has been standardized as IEEE 802.16.

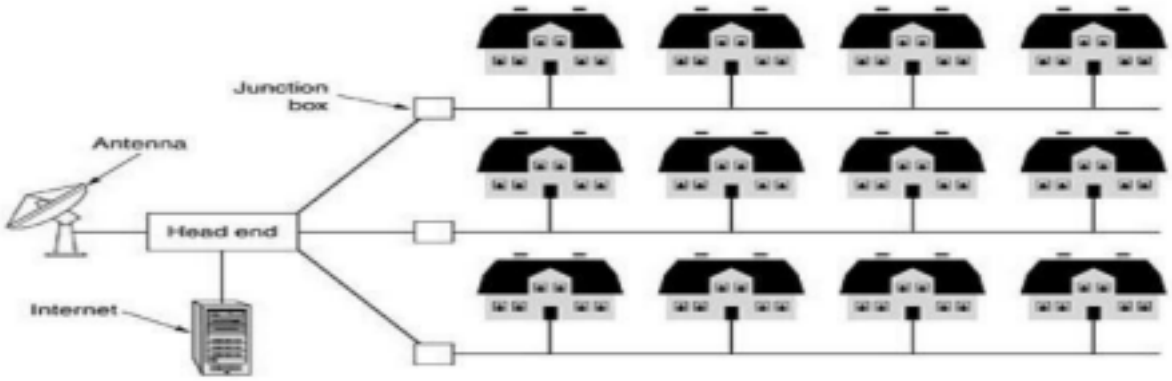


Fig.2: Metropolitan area network based on cable TV.

A MAN is implemented by a standard called DQDB (Distributed Queue Dual Bus) or IEEE 802.16. DQDB has two unidirectional buses (or cables) to which all the computers are attached.

Wide Area Network (WAN).

Wide Area Network:

A wide area network, or WAN, spans a large geographical area, often a country or continent. It contains a collection of machines intended for running user (i.e., application) programs. These machines are called as hosts. The hosts are connected by a communication subnet, or just subnet for short. The hosts are owned by the customers (e.g., people's personal computers), whereas the communication subnet is typically owned and operated by a telephone company or Internet service provider. The job of the subnet is to carry messages from host to host, just as the telephone system carries words from speaker to listener.

Separation of the pure communication aspects of the network (the subnet) from the application aspects (the hosts), greatly simplifies the complete network design. In most wide area networks, the subnet consists of two distinct components: transmission lines and switching elements. Transmission lines move bits between machines. They can be made of copper wire, optical fiber, or even radio links. In most WANs, the network contains numerous transmission lines, each one connecting a pair of routers. If two routers that do not share a transmission line wish to communicate, they must do this indirectly, via other routers. When a packet is sent from one router to another via one or more intermediate routers, the packet is received at each intermediate router in its entirety, stored there until the required output line is free, and then forwarded. A subnet organized according to this principle is called a store-and-forward or packet-switched subnet. Nearly all wide area networks (except those using satellites) have store-and-forward subnets. When the packets are small and all the same size, they are often called cells. The principle of a packet-switched WAN is so important. Generally, when a process on some host has a message to be sent to a process on some other host, the sending host first cuts the message into packets, each one bearing its number in the sequence. These packets are then injected into the network one at a time in quick succession. The packets are transported individually over the network and deposited at the receiving host, where they are reassembled into the original message and delivered to the receiving process. A stream of packets resulting from some initial message is illustrated in Fig.

In this figure, all the packets follow the route ACE, rather than ABDE or ACDE. In some networks all packets from a given message must follow the same route; in others each packed is routed separately. Of course, if ACE is the best route, all packets may be sent along it, even if each packet is individually routed.

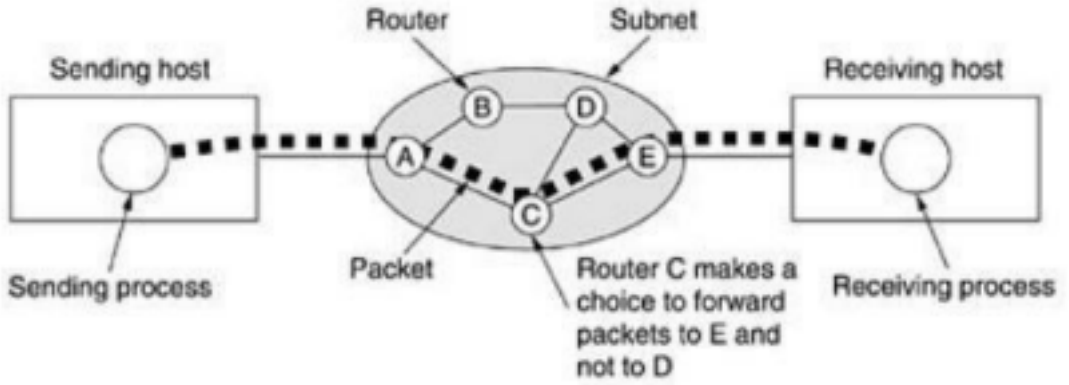


Fig.3.1: A stream of packets from sender to receiver.

Not all WANs are packet switched. A second possibility for a WAN is a satellite system. Each router has an antenna through which it can send and receive. All routers can hear the output from the satellite, and in some cases they can also hear the upward transmissions of their fellow routers to the satellite as well. Sometimes the routers are connected to a substantial point-to-point subnet, with only some of them having a satellite antenna. Satellite networks are inherently broadcast and are most useful when the broadcast property is important.

1.3 THE INTERNET

The Internet has revolutionized many aspects of our daily lives. It has affected the way we do business as well as the way we spend our leisure time. Count the ways you've used the Internet recently. Perhaps you've sent electronic mail (e-mail) to a business associate, paid a utility bill, read a newspaper from a distant city, or looked up a local movie schedule-all by using the Internet. Or maybe you researched a medical topic, booked a hotel reservation, chatted with a fellow Trekkie, or comparison-shopped for a car. The Internet is a communication system that has brought a wealth of information to our fingertips and organized it for our use.

A Brief History

A network is a group of connected communicating devices such as computers and printers. An internet (note the lowercase letter i) is two or more networks that can communicate with each other. The most notable internet is called the Internet (uppercase letter I), a collaboration of more

than hundreds of thousands of interconnected networks. Private individuals as well as various organizations such as government agencies, schools, research facilities, corporations, and libraries in more than 100 countries use the Internet. Millions of people are users. Yet this extraordinary communication system only came into being in 1969.

In the mid-1960s, mainframe computers in research organizations were standalone devices. Computers from different manufacturers were unable to communicate with one another. The Advanced Research Projects Agency (ARPA) in the Department of Defense (DoD) was interested in finding a way to connect computers so that the researchers they funded could share their findings, thereby reducing costs and eliminating duplication of effort.

In 1967, at an Association for Computing Machinery (ACM) meeting, ARPA presented its ideas for ARPANET, a small network of connected computers. The idea was that each host computer (not necessarily from the same manufacturer) would be attached to a specialized computer, called an *inteiface message processor* (IMP). The IMPs, in tum, would be connected to one another. Each IMP had to be able to communicate with other IMPs as well as with its own attached host. By 1969, ARPANET was a reality. Four nodes, at the University of California at Los Angeles (UCLA), the University of California at Santa Barbara (UCSB), Stanford Research Institute (SRI), and the University of Utah, were connected via the IMPs to form a network. Software called the *Network Control Protocol* (NCP) provided communication between the hosts.

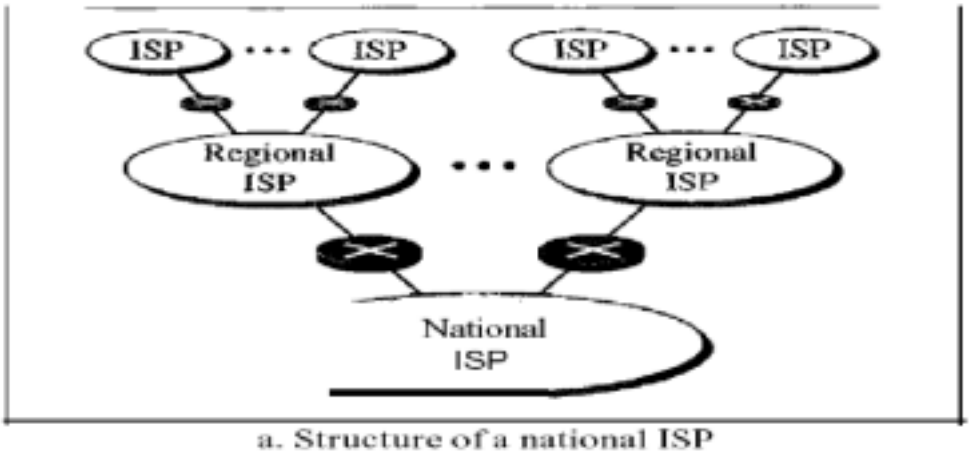
In 1972, Vint Cerf and Bob Kahn, both of whom were part of the core ARPANET group, collaborated on what they called the *Internetting Projec1.* Cerf and Kahn's landmark 1973 paper outlined the protocols to achieve end-to-end delivery of packets. This paper on Transmission Control Protocol (TCP) included concepts such as encapsulation, the datagram, and the functions of a gateway. Shortly thereafter, authorities made a decision to split TCP into two protocols: Transmission Control Protocol (TCP) and Internetworking Protocol (lP). IP would handle datagram routing while TCP would be responsible for higher-level functions such as

segmentation, reassembly, and error detection. The internetworking protocol became known as TCPIIP.

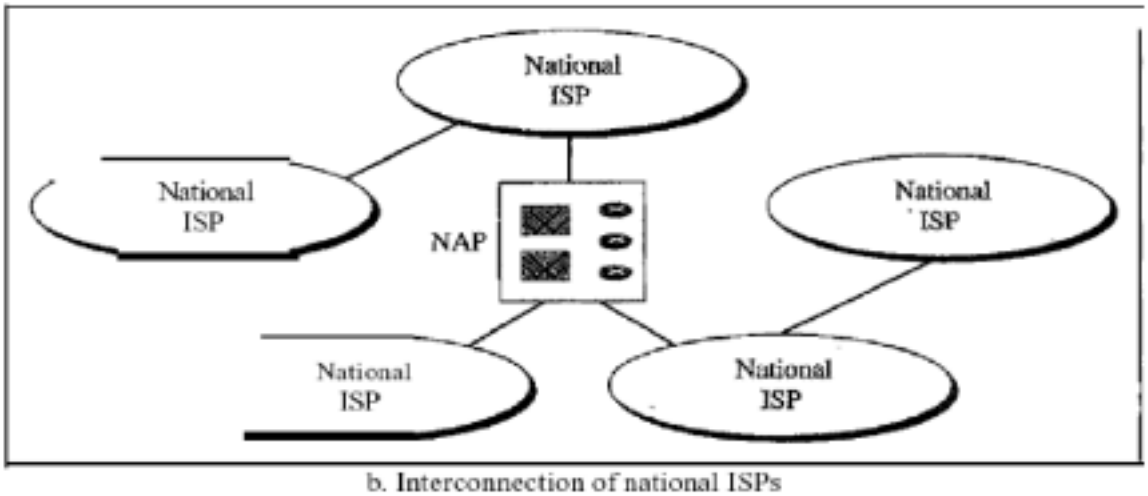
The Internet Today

The Internet has come a long way since the 1960s. The Internet today is not a simple hierarchical

structure. It is made up of many wide- and local-area networks joined by connecting devices and switching stations. It is difficult to give an accurate representation of the Internet because it is continually changing-new networks are being added, existing networks are adding addresses, and networks of defunct companies are being removed. Today most end users who want Internet connection use the services of Internet service providers (lSPs). There are international service providers, national service providers, regional service providers, and local service providers. The Internet today is run by private companies, not the government. Figure 1.13 shows a conceptual (not geographic) view of the Internet.



a. Structure of a national ISP



b. Interconnection of national ISPs

International Internet Service Providers:

At the top of the hierarchy are the international service providers that connect nations together.

National Internet Service Providers:

The national Internet service providers are backbone networks created and maintained by

specialized companies. There are many national ISPs operating in North America; some of the most well known are SprintLink, PSINet, UUNet Technology, AGIS, and internet Mel. To provide connectivity between the end users, these backbone networks are connected by complex switching stations (normally run by a third party) called network access points (NAPs). Some national ISP networks are also connected to one another by private switching stations called *peering points.* These normally operate at a high data rate (up to 600 Mbps).

Regional Internet Service Providers:

Regional internet service providers or regional ISPs are smaller ISPs that are connected to one or more national ISPs. They are at the third level of the hierarchy with a smaller data rate.

Local Internet Service Providers:

Local Internet service providers provide direct service to the end users. The local ISPs can be connected to regional ISPs or directly to national ISPs. Most end users are connected to the local ISPs. Note that in this sense, a local ISP can be a company that just provides Internet services, a corporation with a network that supplies services to its own employees, or a nonprofit organization, such as a college or a university, that runs its own network. Each of these local ISPs can be connected to a regional or national service provider.

1.4 PROTOCOLS AND STANDARDS

Protocols:

In computer networks, communication occurs between entities in different systems. An entity is anything capable of sending or receiving information. However, two entities cannot simply send bit streams to each other and expect to be understood. For communication to occur, the entities must agree on a protocol. A protocol is a set of rules that govern data communications. A protocol defines what is communicated, how it is communicated, and when it is communicated. The key elements of a protocol are syntax, semantics, and timing.

o Syntax. The term *syntax* refers to the structure or format of the data, meaning the order in which they are presented. For example, a simple protocol might expect the first 8 bits of data to be the address of the sender, the second 8 bits to be the address of the receiver, and the rest of the stream to be the message itself.

o Semantics. The word *semantics* refers to the meaning of each section of bits. How is a

particular pattern to be interpreted, and what action is to be taken based on that interpretation? For example, does an address identify the route to be taken or the final destination of the message?

o Timing. The term *timing* refers to two characteristics: when data should be sent and how fast they can be sent. For example, if a sender produces data at 100 Mbps but the receiver can process data at only 1 Mbps, the transmission will overload the receiver and some data will be lost.

Standards

Standards are essential in creating and maintaining an open and competitive market for equipment manufacturers and in guaranteeing national and international interoperability of data and telecommunications technology and processes. Standards provide guidelines to manufacturers, vendors, government agencies, and other service providers to ensure the kind of interconnectivity necessary in today's marketplace and in international communications.

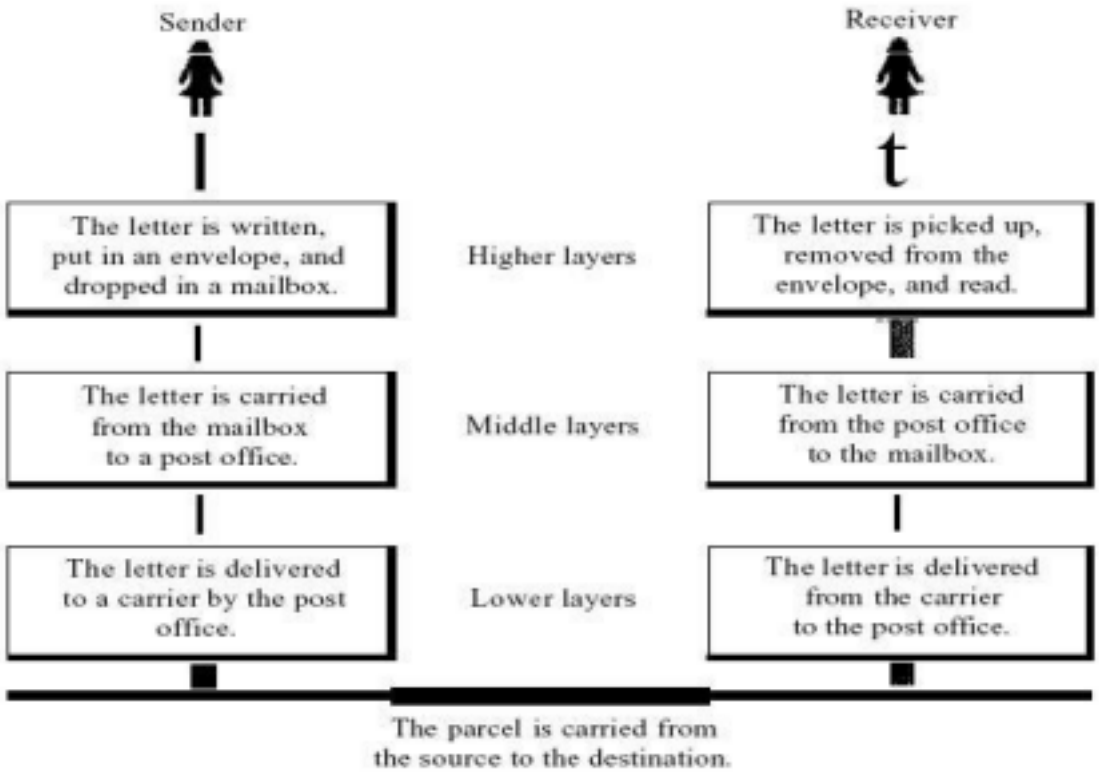
Data communication standards fall into two categories: *de facto* (meaning "by fact" or "by convention") and *de jure* (meaning "by law" or "by regulation").

o De facto. Standards that have not been approved by an organized body but have been adopted as standards through widespread use are de facto standards. De facto standards are often established originally by manufacturers who seek to define the functionality of a new product or technology.

o De jure. Those standards that have been legislated by an officially recognized body are de jure standards.

1.5 LAYERED TASKS

We use the concept of layers in our daily life. As an example, let us consider two friends who communicate through postal maiL The process of sending a letter to a friend would be complex if there were no services available from the post office. Below Figure shows the steps in this task.



Sender, Receiver, and Carrier

In Figure we have a sender, a receiver, and a carrier that transports the letter. There is a hierarchy of tasks.

At the Sender Site

Let us first describe, in order, the activities that take place at the sender site. o Higher layer. The sender writes the letter, inserts the letter in an envelope, writes the sender and receiver addresses, and drops the letter in a mailbox.

o Middle layer. The letter is picked up by a letter carrier and delivered to the post office. o Lower layer. The letter is sorted at the post office; a carrier transports the letter. *On the Way:* The letter is then on its way to the recipient. On the way to the recipient's local post office, the letter may actually go through a central office. In addition, it may be transported by truck, train, airplane, boat, or a combination of these.

At the Receiver Site

o Lower layer. The carrier transports the letter to the post office.

o Middle layer. The letter is sorted and delivered to the recipient's mailbox. o Higher layer. The receiver picks up the letter, opens the envelope, and reads it. **1.6 The**

OSI Reference Model:

The OSI model (minus the physical medium) is shown in Fig. This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It was revised in 1995(Day, 1995). The model is called the ISO-OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems.

The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

1. A layer should be created where a different abstraction is needed.
2. Each layer should perform a well-defined function.
3. The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
4. The layer boundaries should be chosen to minimize the information flow across the interfaces.
5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

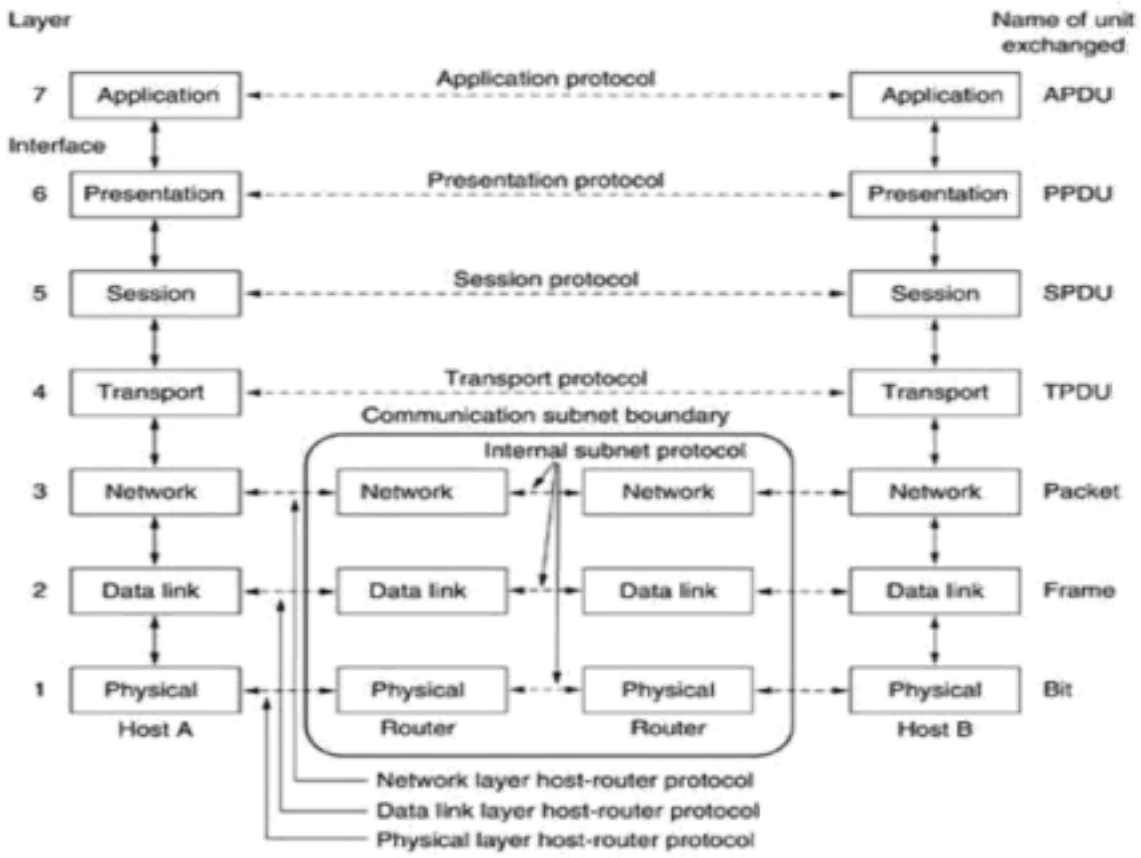


Fig.4: The OSI reference model

The Physical Layer:

The physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not as a 0 bit.

The Data Link Layer:

The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors to the network layer. It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmits the frames sequentially. If the service is reliable, the receiver confirms correct receipt of each frame by sending back an acknowledgement frame.

Another issue that arises in the data link layer (and most of the higher layers as well) is how to keep a fast transmitter from drowning a slow receiver in data. Some traffic regulation mechanism is often needed to let the transmitter know how much buffer space the receiver has at the moment. Frequently, this flow regulation and the error handling are integrated.

The Network Layer:

The network layer controls the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes can be based on static tables that are "wired into" the network and rarely changed. They can also be determined at the start of each conversation, for example, a terminal session (e.g., a login to a remote machine). Finally, they can be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in one another's way, forming bottlenecks. The control of such congestion also belongs to the network layer. More generally, the quality of service provided (delay, transit time, jitter, etc.) is also a network layer issue.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these problems to allow heterogeneous networks to be interconnected. In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

The Transport Layer:

The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the upper layers from the inevitable changes in the hardware technology. The transport layer also determines what type of service to provide to the session layer, and, ultimately, to the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages or bytes in the order in which they were sent. However, other possible kinds of transport service are the transporting of isolated messages, with no guarantee about the order of delivery, and the broadcasting of messages to multiple destinations. The type of service is determined when the connection is established.

The transport layer is a true end-to-end layer, all the way from the source to the destination. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages. In the lower layers,

the protocols are between each machine and its immediate neighbours, and not between the ultimate source and destination machines, which may be separated by many routers.

The Session Layer:

The session layer allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation at the same time), and synchronization (check pointing long transmissions to allow them to continue from where they were after a crash).

The Presentation Layer:

The presentation layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire." The presentation layer manages these abstract data structures and allows higher-level data structures (e.g., banking records), to be defined and exchanged.

The Application Layer:

The application layer contains a variety of protocols that are commonly needed by users. One widely-used application protocol is HTTP (Hypertext Transfer Protocol), which is the basis for the World Wide Web. When a browser wants a Web page, it sends the name of the page it wants to the server using HTTP. The server then sends the page back. Other application protocols are used for file transfer, electronic mail, and network news.

1.7 The TCP/IP Reference Model:

The TCP/IP reference model was developed prior to OSI model. The major design goals of this model were,

1. To connect multiple networks together so that they appear as a single network. 2. To survive after partial subnet hardware failures.
3. To provide a flexible architecture.

Unlike OSI reference model, TCP/IP reference model has only 4 layers. They are, 1. Host-to-Network Layer

2. Internet Layer

3. Transport Layer

4. Application Layer

Application Layer

Transport Layer

Internet Layer Host-to

Network Layer

Host-to-Network Layer:

The TCP/IP reference model does not really say much about what happens here, except to point out that the host has to connect to the network using some protocol so it can send IP packets to it. This protocol is not defined and varies from host to host and network to network. **Internet Layer:**

This layer, called the internet layer, is the linchpin that holds the whole architecture together. Its job is to permit hosts to inject packets into any network and have they travel independently to the destination (potentially on a different network). They may even arrive in a different order than they were sent, in which case it is the job of higher layers to rearrange them, if in-order delivery is desired. Note that "internet" is used here in a generic sense, even though this layer is present in the Internet.

The internet layer defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go. Packet routing is clearly the major issue here, as is avoiding congestion. For these reasons, it is reasonable to say that the TCP/IP internet layer is similar in functionality to the OSI network layer. Fig. shows this correspondence.

The Transport Layer:

The layer above the internet layer in the TCP/IP model is now usually called the transport layer. It is designed to allow peer entities on the source and destination hosts to carry on a conversation, just as in the OSI transport layer. Two end-to-end transport protocols have been defined here. The first one, TCP (Transmission Control Protocol), is a reliable connection oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. It fragments the incoming byte stream into discrete messages and passes each one on to the internet layer. At the destination, the receiving TCP process reassembles the received messages into the output stream. TCP also handles flow control

to make sure a fast sender cannot swamp a slow receiver with more messages than it can handle.

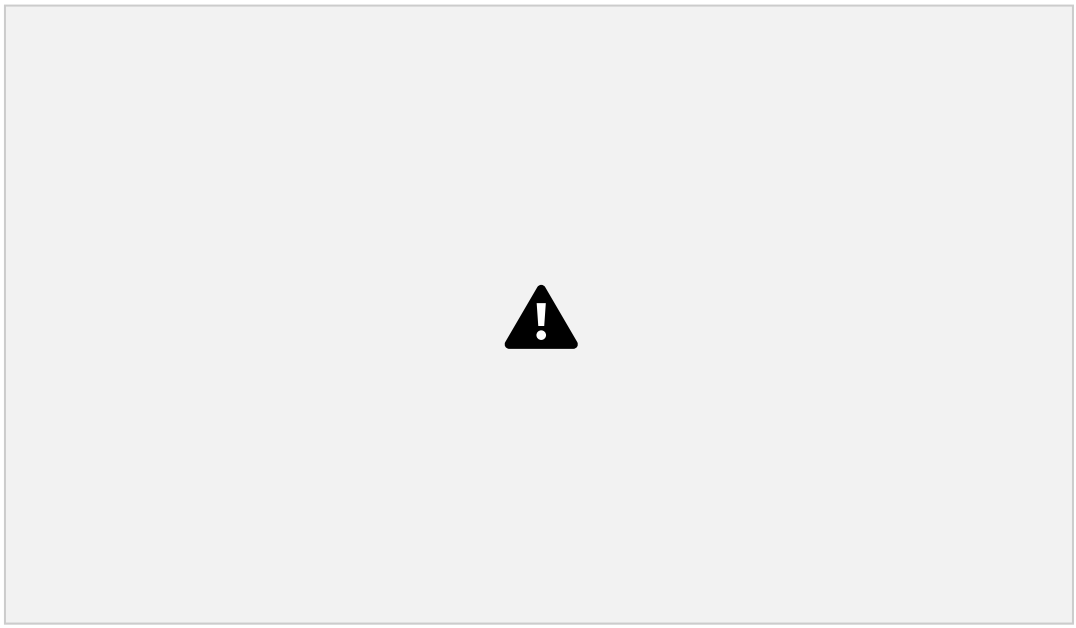


Fig.1: The TCP/IP reference model.

The second protocol in this layer, UDP (User Datagram Protocol), is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own. It is also widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery, such as transmitting speech or video. The relation of IP, TCP, and UDP is shown in Fig.2. Since the model was developed, IP has been implemented on many other networks.

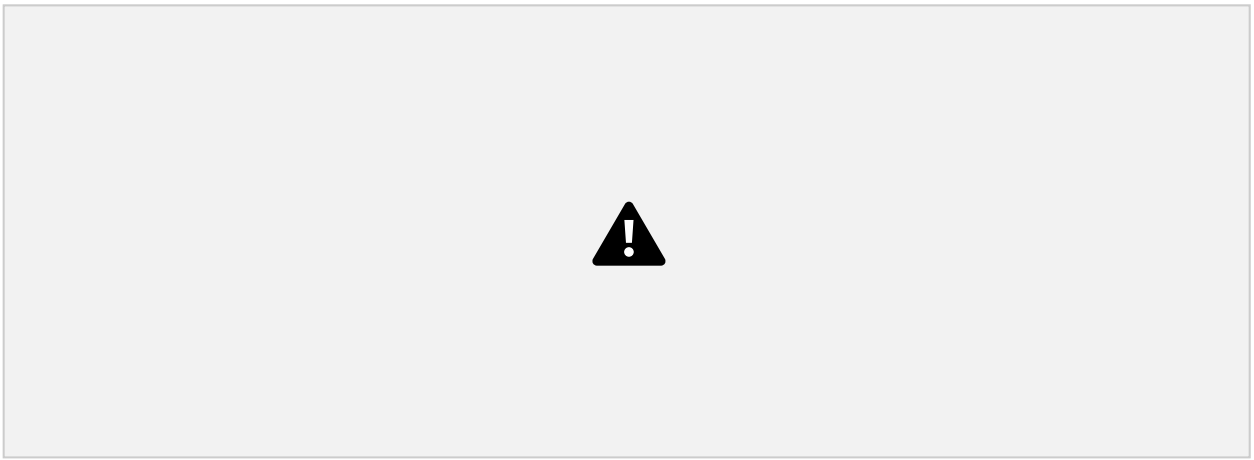


Fig.2: Protocols and networks in the TCP/IP model initially.

The Application Layer:

The TCP/IP model does not have session or presentation layers. On top of the transport layer is the application layer. It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP), as shown in Fig.6.2. The virtual terminal protocol allows a user on one machine to log onto a distant machine and work there. The file transfer protocol provides a way to move data efficiently from one machine to another. Electronic mail was originally just a kind of file transfer, but later a specialized protocol (SMTP) was developed for it. Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses, NNTP, the protocol for moving USENET news articles around, and HTTP, the protocol for fetching pages on the World Wide Web, and many others.

Comparison of the OSI and TCP/IP Reference Models:

The OSI and TCP/IP reference models have much in common. Both are based on the concept of a stack of independent protocols. Also, the functionality of the layers is roughly similar. For example, in both models the layers up through and including the transport layer are there to provide an end-to-end, network-independent transport service to processes wishing to communicate. These layers form the transport provider. Again in both models, the layers above transport are application-oriented users of the transport service. Despite these fundamental similarities, the two models also have many differences Three concepts are central to the OSI model:

1. Services.
2. Interfaces.
3. Protocols.

Probably the biggest contribution of the OSI model is to make the distinction between these three concepts explicit. Each layer performs some services for the layer above it. The service definition tells what the layer does, not how entities above it access it or how the layer works. It defines the layer's semantics.

A layer's interface tells the processes above it how to access it. It specifies what the parameters are and what results to expect. It, too, says nothing about how the layer works inside.

Finally, the peer protocols used in a layer are the layer's own business. It can use any protocols it wants to, as long as it gets the job done (i.e., provides the offered services). It can also change them at will without affecting software in higher layers.

The TCP/IP model did not originally clearly distinguish between service, interface, and protocol, although people have tried to retrofit it after the fact to make it more OSI-like. For example, the only real services offered by the internet layer are SEND IP PACKET and RECEIVE IP PACKET.

As a consequence, the protocols in the OSI model are better hidden than in the TCP/IP model and can be replaced relatively easily as the technology changes. Being able to make such changes is one of the main purposes of having layered protocols in the first place. The OSI reference model was devised before the corresponding protocols were invented. This ordering means that the model was not biased toward one particular set of protocols, a fact that made it quite general. The downside of this ordering is that the designers did not have much experience with the subject and did not have a good idea of which functionality to put in which layer.

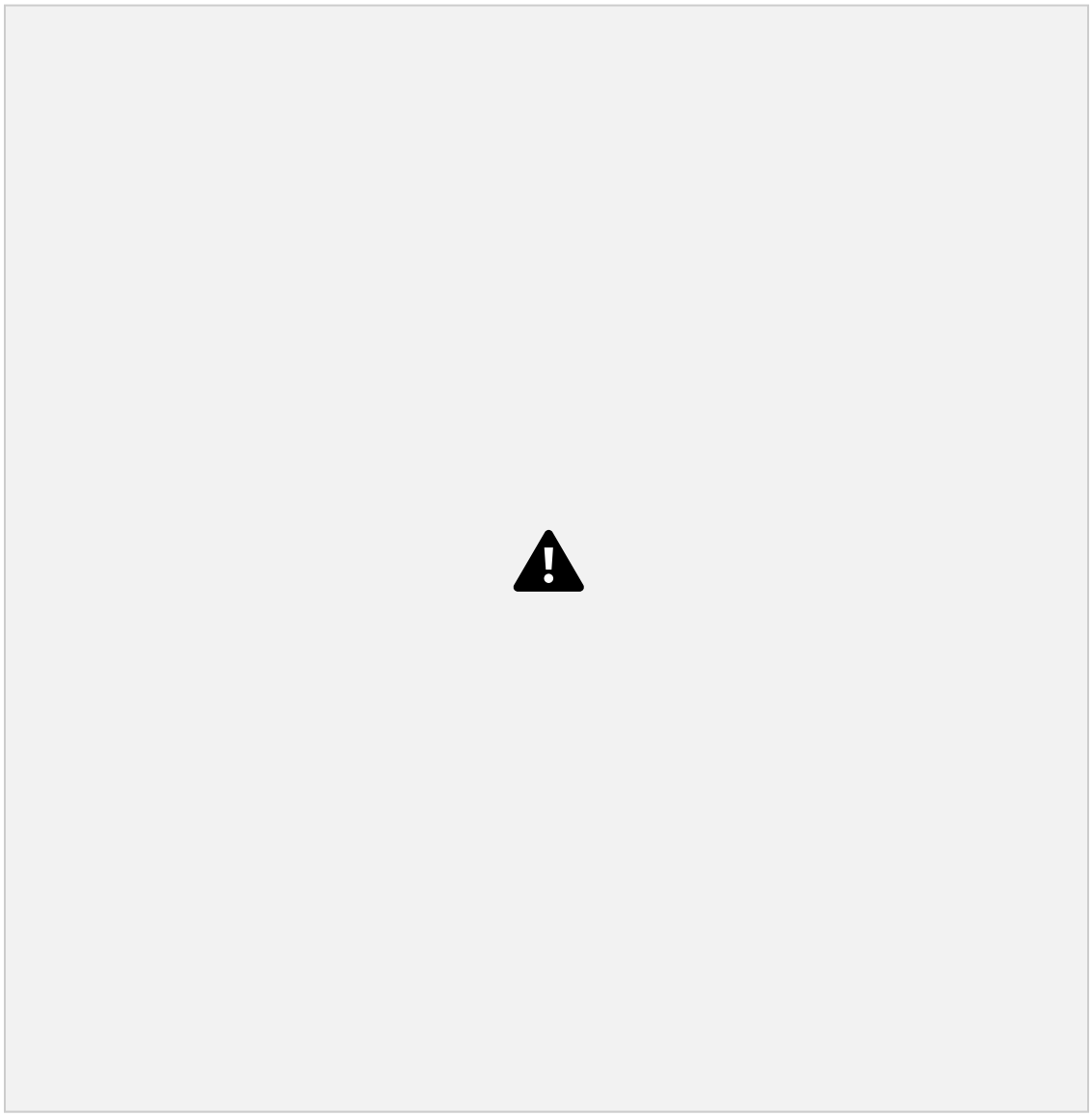
Another difference is in the area of connectionless versus connection-oriented communication. The OSI model supports both connectionless and connection-oriented communication in the network layer, but only connection-oriented communication in the transport layer, where it counts (because the transport service is visible to the users). The TCP/IP model has only one mode in the network layer (connectionless) but supports both modes in the transport layer, giving the users a choice. This choice is especially important for simple request-response protocols.

OSI(Open System Interconnection)	**TCP/IP(Transmission Control Protocol / Internet Protocol)**
1. OSI is a generic, protocol independent standard, acting as a communication gateway between the network and end user.	1. TCP/IP model is based on standard protocols around which the Internet has developed. It is a communication protocol, which allows connection of hosts over a network.
2. In OSI model the transport layer guarantees the delivery of packets.	2. In TCP/IP model the transport layer does not guarantees delivery of packets. Still the TCP/IP model is more reliable.
3. Follows vertical approach.	3. Follows horizontal approach.
4. OSI model has a separate Presentation layer and Session layer.	4. TCP/IP does not have a separate Presentation layer or Session layer.
5. Transport Layer is Connection Oriented.	5. Transport Layer is both Connection Oriented and Connection less.

6. Network Layer is both Connection Oriented and Connection less.	6. Network Layer is Connection less.
7. OSI is a reference model around which the networks are built. Generally it is used as a guidance tool.	7. TCP/IP model is, in a way implementation of the OSI model.
8. Network layer of OSI model provides both connection oriented and connectionless service.	8. The Network layer in TCP/IP model provides connectionless service.
9. OSI model has a problem of fitting the protocols into the model.	9. TCP/IP model does not fit any protocol
10. Protocols are hidden in OSI model and are easily replaced as the technology topology	10. In TCP/IP replacing protocol is not easy.

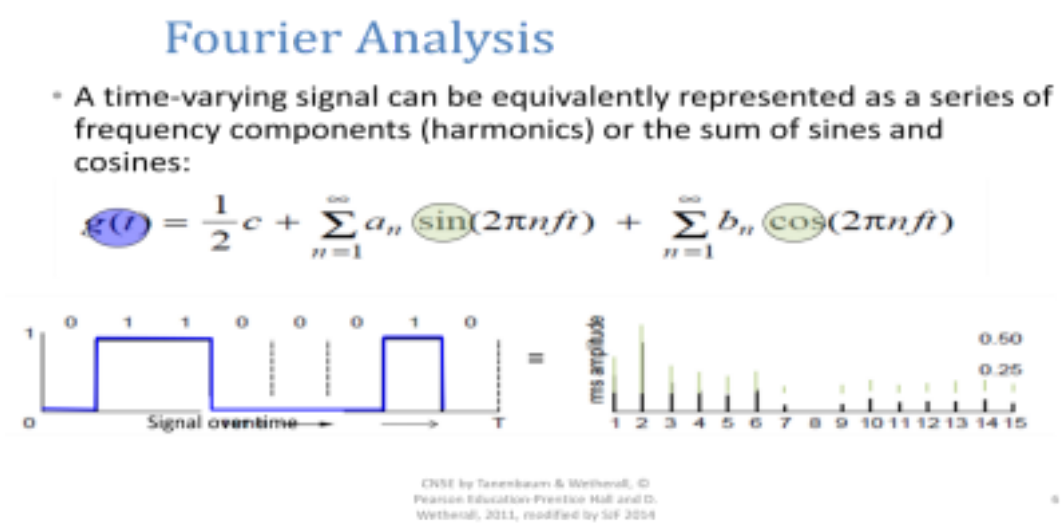
changes.	
11. OSI model defines services, interfaces and protocols very clearly and makes clear distinction between them. It is protocol independent.	11. In TCP/IP, services, interfaces and protocols are not clearly separated. It is also protocol dependent.
12. It has 7 layers	12. It has 4 layers

Diagrammatic Comparison between OSI Reference Model and TCP/IP Reference Model



UNIT -2

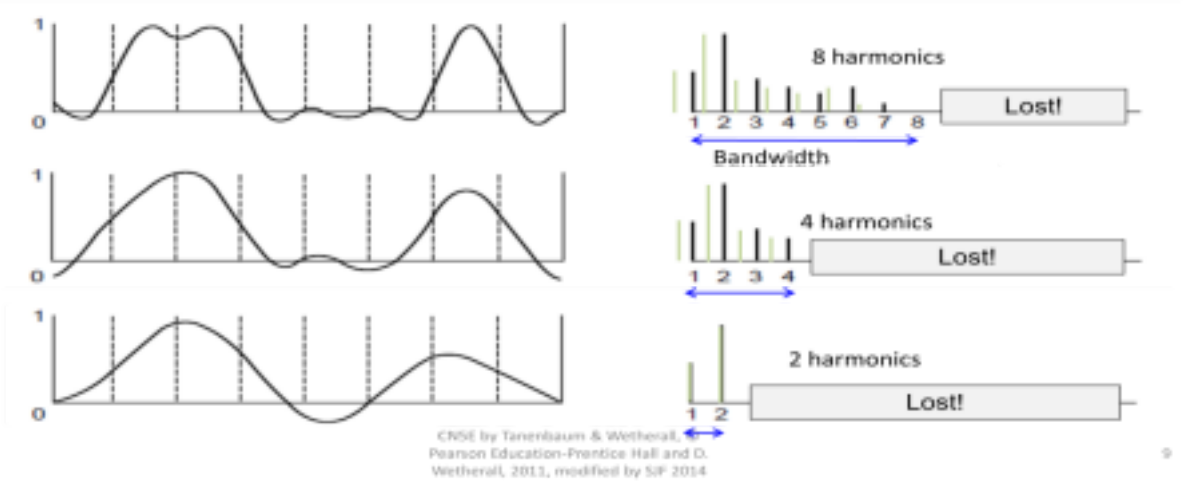
Physical Layer:

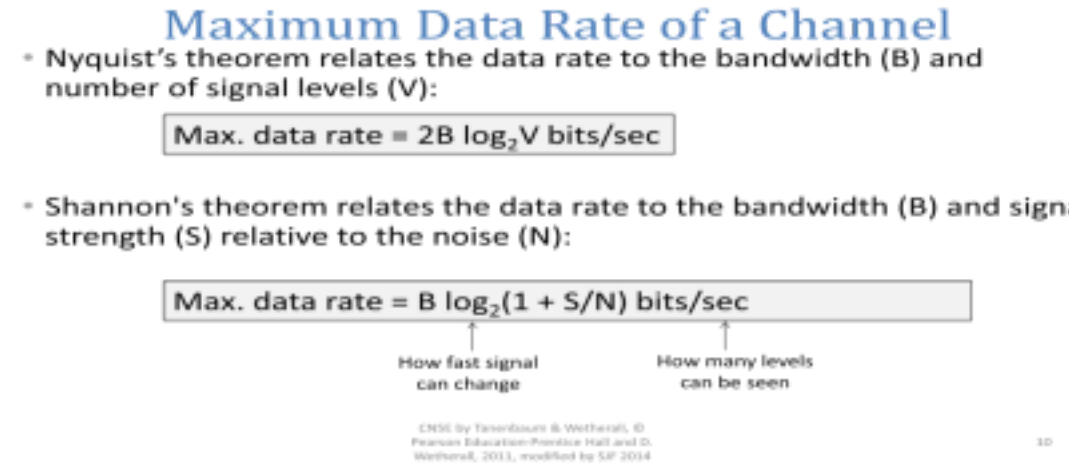


- Transmitted signals lose some power as they are transmitted.
- For a wire, amplitudes are transmitted mostly undiminished from 0 up to some frequency f. Frequencies above this frequency f are attenuated (reduced).
- The width of this frequency range is called the **bandwidth**.
    - Baseband run from 0 to some max frequency
    - Passband- shifted to occupy higher frequencies such as wireless

Bandwidth

- Bandwidth is a physical property of the transmission medium such as the construction, thickness and length of the wire or fiber.
- Limiting the bandwidth, limits the data rate.
- Goal for digital transmission is to receive a signal with enough fidelity to reconstruct the sequence of bits that was sent.
- Bandwidth – to Electrical Engineers –(analog) means – a quantity measured in Hz (cycles per second)
- Bandwidth – to Computer Scientists – (digital) means – the maximum data rate of a channel (bits/second)





Transmission Media

Transmission Media – Guided

There are several types of cable which are commonly used with LANs. In some cases, a network will utilize only one type of cable, other networks will use a variety of cable types. The type of cable chosen for a network is related to the network's topology, protocol, and size. Understanding the characteristics of different types of cable and how they relate to other aspects of a network is necessary for the development of a successful network.

- Unshielded Twisted Pair (UTP) Cable
- Shielded Twisted Pair (STP) Cable
- Coaxial Cable
- Fiber Optic Cable

Unshielded Twisted Pair (UTP) Cable

Twisted pair cabling comes in two varieties: shielded and unshielded. Unshielded twisted pair (UTP) is the most popular and is generally the best option for school networks.

Figure 2.1 Unshielded Twisted Pair

The quality of UTP may vary from telephone-grade wire to extremely high-speed cable. The cable has four pairs of wires inside the jacket. Each pair is twisted with a different number of twists per inch to help eliminate interference from adjacent pairs and other electrical devices.

The tighter the twisting, the higher the supported transmission rate and the greater the cost per foot. The EIA/TIA (Electronic Industry Association / Telecommunication Industry Association) has established standards of UTP and rated five categories of wire.

Type	**Use**
Category 1	Voice Only (Telephone Wire)
Category 2	Data to 4 Mbps (LocalTalk)
Category 3	Data to 10 Mbps (Ethernet)
Category 4	Data to 20 Mbps (16 Mbps Token Ring)
Category 5	Data to 100 Mbps (Fast Ethernet)

Table Categories of Unshielded Twisted Pair

Unshielded Twisted Pair Connector

The standard connector for unshielded twisted pair cabling is an RJ-45 connector. This is a

plastic connector that looks like a large telephone-style connector (See fig. 2.2). a slot allows the RJ-45 to be inserted only one way. RJ stands for Registered Jack, implying that the connector follows a standard borrowed from the telephone industry. This standard designates which wire goes with each pin inside the connector.

Figure 2.2 RJ-45

Shielded Twisted Pair (STP) Cable

A disadvantage of UTP is that it may be susceptible to radio and electrical frequency interference. Shielded twisted pair |(STP) is suitable for environments with electrical interference; however, the extra shielding can make the cables quite bulky. Shielded twisted pair is often used on networks using Token Ring topology.

Coaxial Cable

Coaxial cabling has a single copper conductor at its center. A plastic layer provides insulation between the center conductor and the braided metal shield (See fig. 3). The metal shield helps to block any outside interference from fluorescent lights, motors, and other computers.

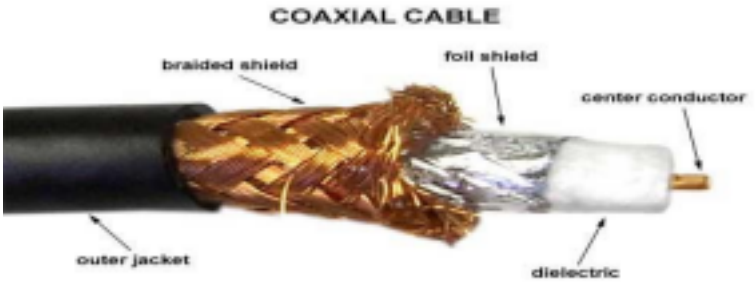


Fig. 2.3 Coaxial cable

Although coaxial cabling is difficult to install, it is highly resistant to signal interference. In addition, it can support grater cable lengths between network devices than twisted pair cable. The two types of coaxial cabling are thick coaxial and thin coaxial.

Thin coaxial cable is also referred to as thinnet. 10base2 refers to the specifications for thin coaxial cable carrying Ethernet signals. The 2 refers to the approximate maximum segment length being 200 meters. In actual fact the maximum segment length is 185 meters. Thin coaxial cable is popular in school networks, especially linear bus networks.

Thick coaxial cable is also referred to as thicknet. 10base refers to the specifications for thick coaxial cable carrying Ethernet signals. The 5 refers to the maximum segment length being 500 meters. Thick coaxial cable has an extra protective plastic cover that helps keep moisture away from the center conductor. This makes thick coaxial a great choice when running longer lengths in a linear bus network. One disadvantage of thick coaxial is that it does not bend easily and is difficult to install.

2.1.3 Fiber Optic Cable

Fiber optic cabling consists of a center glass core surrounded by several layers of protective materials. It transmits light rather than electronic signals eliminating the problem of electrical interference.

This makes it ideal for certain environments that contain a large amount of electrical interference. It has also made it the standard for connecting networks between buildings, due to its immunity to the effects of moisture and lighting.

Fiber optic cable has the ability to transmit signals over much longer distances than coaxial and twisted pair. It also has made it the standard for connecting networks between buildings, due to its immunity to the effects of moisture and lighting.

Fiber optic cable has the ability to transmit signals over mush longer distances than coaxial and twisted pair. It also has the capability to carry information at vastly greater speeds. This capacity broadens communication possibilities to include services such as video conferencing and interactive services. The cost of fiber optic cabling is comparable to copper cabling; however it is more difficult to install and modify. 10BaseF refers to the specifications for fiber optic cable carrying Ethernet signals.

Fig. 2.5 Fiber Optic Cable

Facts about fiber optic cables:

- Outer insulating jacket is made of Teflon or PVC.
- Kevlar fiber helps to strengthen the cable and prevent breakage.
- A plastic coating is used to cushion the fiber center.
- Center (core) is made of glass or plastic fibers.

Fiber Optic Connector

The most common connector used with fiber optic cable is an ST connector. It is barrel shaped, similar to a BNC connector. A newer connector, the SC, is becoming more popular. It has a squared face and is easier to connect in a confined space.

Specification	Cable Type	Maximum length
10BaseT	Unshielded Twisted Pair	100 meters
10Base2	Thin Coaxial	185 meters
10Base5	Thick Coaxial	500 meters
10BaseF	Fiber Optic	2000 meters
100BaseT	Unshielded Twisted Pair	100 meters
100BaseTX	Unshielded Twisted Pair	220 meters

Table 2.2 Ethernet Cable Summary

2.2 Transmission Media – Unguided

Unguided transmission media is data signals that flow through the air. They are not guided or bound to a channel to follow.

Unguided media transport electromagnetic waves without using a physical conductor. This type of communication is often referred to as wireless communication. Signals are normally broadcast through free space and thus are available to anyone who has a device receiving them. Unguided signals can travel from the source to destination in several ways: ground propagation, sky propagation, and line-of-sight propagation.

In **ground propagation**, radio waves travel through the lowest portion of the atmosphere,

hugging the earth. These low-frequency signals emanate in all directions from the transmitting antenna and follow the curvature of the planet. Distance depends on the amount of power in the signal: *The greater the power, the greater the distance.* Ground waves have carrier frequencies up to 2 MHz. AM radio is an example of ground wave propagation.

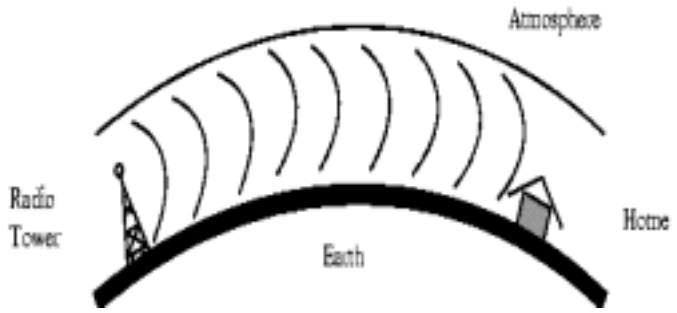


Fig. 2.6 Ground Wave Propagation

In **sky propagation**, higher frequency radio waves radiate upward into the ionosphere (the layer of atmosphere where the particles exist as ions) where they are reflected back to the earth. This type of transmission allows for greater distances with lower output power.

It is sometimes called double hop propagation. It operates in the frequency range of 30 – 85 MHz. Because it depends on the earth's ionosphere, it changes with the weather and time of day. The signal bounces off of the ionosphere and back to the earth. Ham radios operate in this range. Other books called this **Ionospheric propagation**.

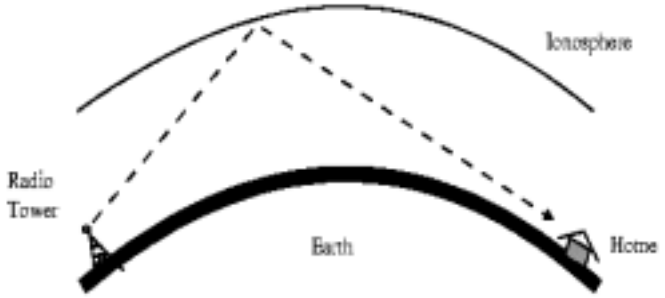


Fig. 2.7 Ionospheric Propagation

In **line-of-sight propagation**, very high-frequency signals are transmitted in straight lines directly from antenna to antenna. Antennas must be directional, facing each other and either tall enough or close enough together not to be affected by the curvature the earth. Line-of sight propagation is tricky because radio transmission cannot be completely focused.

It is sometimes called space waves or tropospheric propagation. It is limited by the curvature of the earth for ground-based stations (100 km, from horizon to horizon). Reflected waves can cause promlems. Axamples are: FM radio, microwave and satellite.

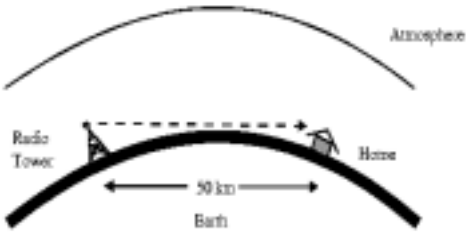


Fig. 2.8 Line-of-sight Propagation

The section of the electromagnetic spectrum defined as radio waves and microwaves is divided into eight ranges, called bands, each regulated by government authorities. These bands are rated from very low frequency (VLF) to extremely high frequency (EHF).

Band	*Range*	*Propagation*	*Application*
VLF (very low frequency)	3-30 kHz	Ground	Long-range radio navigation
LF (low frequency)	30-300 kHz	Ground	Radio beacons and navigation locator
MF (middle frequency)	300 kHz-3 MHz	Sky	AM radio
HF (high frequency)	3-30MHz	Sky	Citizens band (CB), ship/aircraft communication
VHF (very high frequency)	30-300MHz	Sky and line-of sight	VHF TV, FM radio
UHF (ultrahigh frequency)	300 MHz-3 GHz	Line-of-sight	UHF TV, cellular phones, paging, satellite
SHF (superhigh frequency)	3-30 GHz	Line –of-sight	Satellite communication
EHF (extremely high frequency)	30-300 GHz	Line-of-sight	Radar, satellite

Table 2.3 Bands

We can divide wireless transmission into three broad groups: radio waves, microwaves, and infrared waves.

2.2.1 Radio Waves

Electromagnetic waves ranging in frequencies between 3 kHz and 1 GHz are normally called radio waves.

Radio waves are omnidirectional. When antenna transmits radio waves, they are propagated in all directions. This means that the sending and receiving antennas do not have to be aligned. A sending antenna sends waves that can be received by any receiving antenna.

The omnidirectional property has a disadvantage too. The radio waves transmitted by one antenna are susceptible to interference by another antenna that may send signals using the same frequency or band.

Radio waves, particularly those of low and medium frequencies, can penetrate walls. This characteristic can be both an advantage and disadvantage. It is an advantage because, for example, an AM radio can receive signals inside a building. It is a disadvantage because we cannot isolate a communication to just inside or outside a building.

2.2.2 Microwaves

Electromagnetic waves having frequencies between 1 and 300 GHz are called microwaves.

Microwaves are unidirectional. When an antenna transmits microwave waves, they can be narrowly focused. This means that the sending and receiving antennas need to be aligned. The unidirectional property has an obvious advantage. A pair of antennas can be aligned without interfering with another pair of aligned antennas. The following describes some characteristics of microwave propagation:

- Microwave propagation is line-of-sight. Since towers with the mounted antennas need to be in direct sight of each other. This also set a limit on the distance between stations depending on the local geography. Towers that are far apart need to be very tall. The curvature of the earth as well as other blocking obstacles does not allow two short towers to communicate by using microwaves. Typically the line of sight due to the Earth's curvature is only 50 km to the horizon. Repeaters are often needed for long-distance communication.
- Very high frequency microwaves cannot penetrate walls. This characteristic can be a disadvantage if receivers are inside the buildings.
- The microwave band is relatively wide, almost 299 GHz. Therefore wider subbands can be assigned, and a high data rate is possible.
- Use of certain portions of the band requires permission from authorities.

2.2.3 Infrared Waves

Infrared waves, with frequencies from 300 GHz to 400 THz (wavelengths from 1 mm to 770 mm), can be used for short-range communication. Infrared waves, having high frequencies, cannot penetrate walls. This advantageous characteristic prevents interference between one system and another; a short-range communication system in one room cannot be affected by another system in the next room. When we use our infrared remote control, we do not

interfere with the use of the remote of our neighbors. However, this same characteristic makes infrared signals useless for long-range communication. In addition, we cannot use infrared waves outside a building because the sun's rays contain infrared waves that can interfere with the communication.

2.2.4 Satellite

Satellites are transponders (units that receive on one frequency and retransmit on another) that are set in geostationary orbits directly over the equator. These geostationary orbits are 36, 000 km from the Earths's surface. At this point, the gravitational pull of the Earth and the centrifugal force of Earth's rotation are balanced and cancel each other out. Centrifugal force is the rotational force placed on the satellite that wants to fling it out into the space.

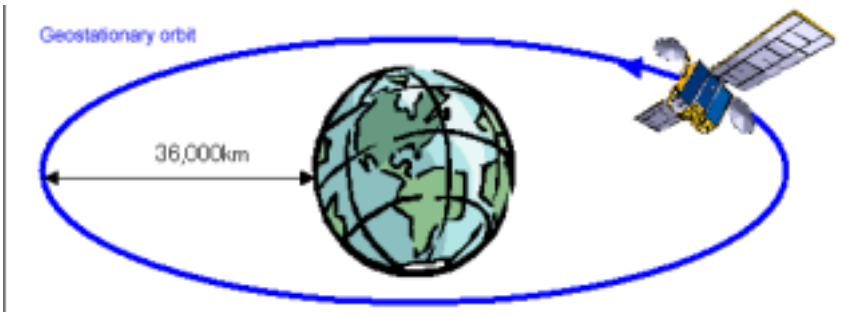


Fig. 2.9 Satellite Communication

The uplink is the transmitter of data to the satellite. The downlink is the receiver of data. Uplinks and downlinks are also called Earth stations because they are located on the Earth. The footprint is the "shadow" that the satellite can transmit to, the shadow being the area that can receive the satellite's transmitted signal.

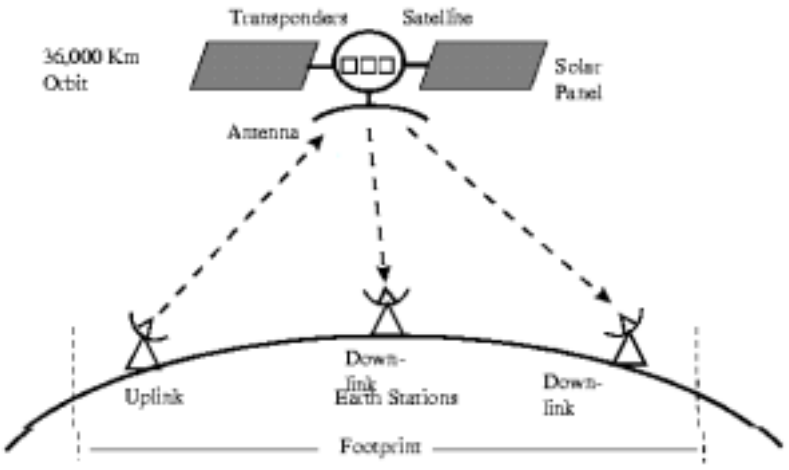


Fig. 2.10 Uplink and Downlink

Digital Modulation techniques

Digital-to-Analog signals is the next conversion we will discuss in this chapter. These

techniques are also called as **Digital Modulation techniques**.

Digital Modulation provides more information capacity, high data security, quicker system availability with great quality communication. Hence, digital modulation techniques have a greater demand, for their capacity to convey larger amounts of data than analog modulation techniques.

There are many types of digital modulation techniques and also their combinations, depending upon the need. Of them all, we will discuss the prominent ones.

ASK –Amplitude Shift Keying

The amplitude of the resultant output depends upon the input data whether it should be a zero level or a variation of positive and negative, depending upon the carrier frequency.

FSK – Frequency Shift Keying

The frequency of the output signal will be either high or low, depending upon the input data applied.

PSK – Phase Shift Keying

The phase of the output signal gets shifted depending upon the input. These are mainly of two types, namely Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK), according to the number of phase shifts. The other one is Differential Phase Shift Keying (DPSK) which changes the phase according to the previous value.

M-ary Encoding

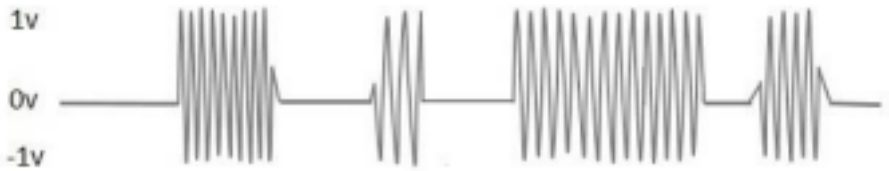
M-ary Encoding techniques are the methods where more than two bits are made to transmit simultaneously on a single signal. This helps in the reduction of bandwidth.

The types of M-ary techniques are –

- M-ary ASK
- M-ary FSK
- M-ary PSK
- **Amplitude Shift Keying (ASK)** is a type of Amplitude Modulation which represents the binary data in the form of variations in the amplitude of a signal.
- Any modulated signal has a high frequency carrier. The binary signal when ASK modulated, gives a **zero** value for **Low** input while it gives the **carrier output** for **High** input.

- The following figure represents ASK modulated waveform along with its input. •

Input binary sequence

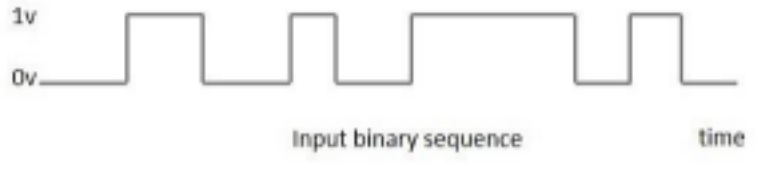


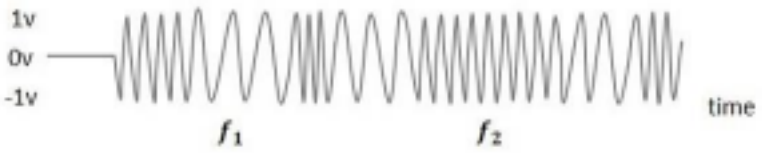
ASK Modulated output wave

Frequency Shift Keying (FSK) is the digital modulation technique in which the frequency of the carrier signal varies according to the digital signal changes. FSK is a scheme of frequency modulation.

The output of a FSK modulated wave is high in frequency for a binary High input and is low in frequency for a binary Low input. The binary **1s** and **0s** are called Mark and Space frequencies.

The following image is the diagrammatic representation of FSK modulated waveform along with its input.

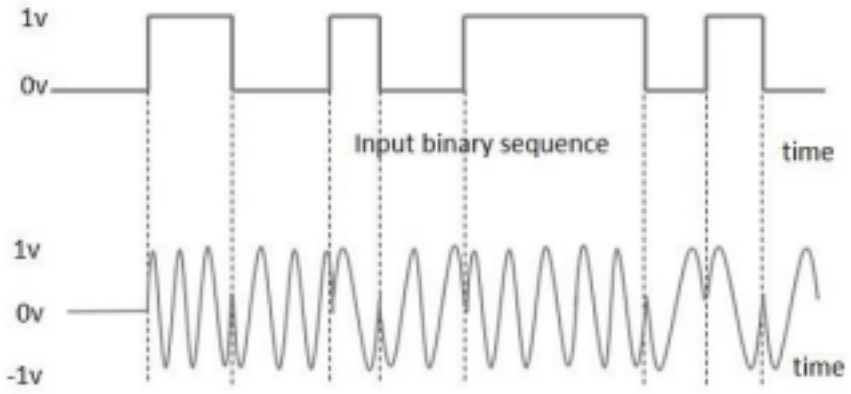




FSK Modulated output wave

Phase Shift Keying (PSK) is the digital modulation technique in which the phase of the carrier signal is changed by varying the sine and cosine inputs at a particular time. PSK technique is widely used for wireless LANs, bio-metric, contactless operations, along with RFID and Bluetooth communications.

Following is the diagrammatic representation of BPSK Modulated output wave along with its given input.



BPSK Modulated output wave

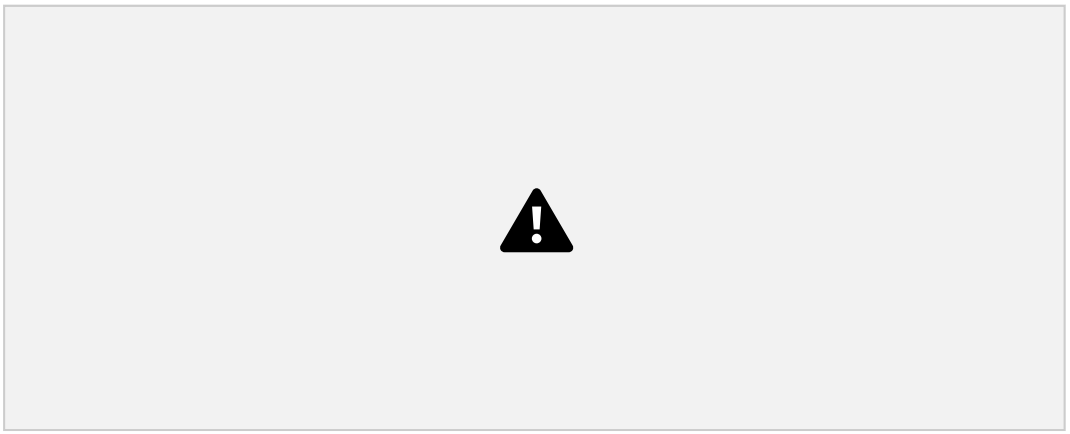
Multiplexing is a technique by which different analog and digital streams of transmission can be simultaneously processed over a shared link. Multiplexing divides the high capacity medium into low capacity logical medium which is then shared by different streams.

Communication is possible over the air (radio frequency), using a physical media (cable), and light (optical fiber). All mediums are capable of multiplexing.

When multiple senders try to send over a single medium, a device called Multiplexer divides the physical channel and allocates one to each. On the other end of communication, a De multiplexer receives data from a single medium, identifies each, and sends to different receivers.

Frequency Division Multiplexing

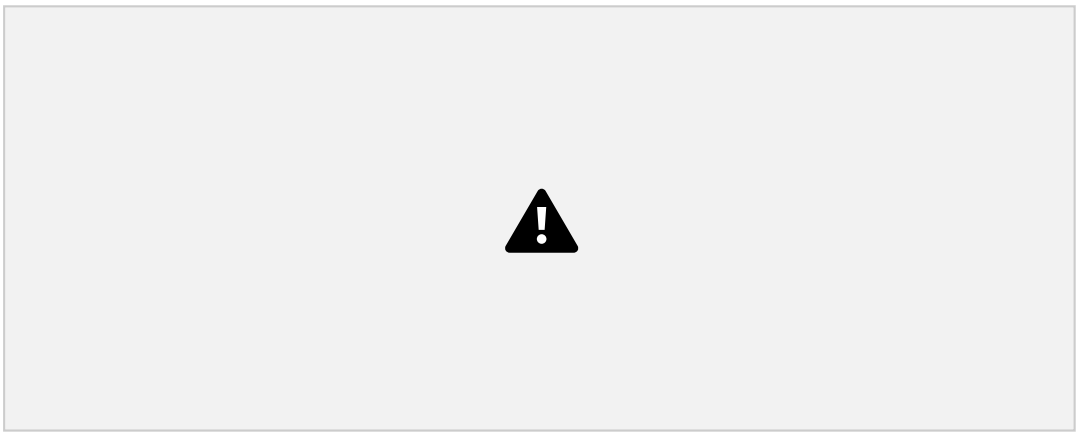
When the carrier is frequency, FDM is used. FDM is an analog technology. FDM divides the spectrum or carrier bandwidth in logical channels and allocates one user to each channel. Each user can use the channel frequency independently and has exclusive access of it. All channels are divided in such a way that they do not overlap with each other. Channels are separated by guard bands. Guard band is a frequency which is not used by either channel.



Time Division Multiplexing

TDM is applied primarily on digital signals but can be applied on analog signals as well. In TDM the shared channel is divided among its user by means of time slot. Each user can transmit data within the provided time slot only. Digital signals are divided in frames, equivalent to time slot i.e. frame of an optimal size which can be transmitted in given time slot.

TDM works in synchronized mode. Both ends, i.e. Multiplexer and De-multiplexer are timely synchronized and both switch to next channel simultaneously.



When channel A transmits its frame at one end,the De-multiplexer provides media to channel A on the other end.As soon as the channel A's time slot expires, this side switches to channel B. On the other end, the De-multiplexer works in a synchronized manner and provides media to channel B. Signals from different channels travel the path in interleaved manner.

Code Division Multiplexing

Code division multiplexing (CDM) is a multiplexing technique that uses spread spectrum communication. In spread spectrum communications, a narrowband signal is spread over a

larger band of frequency or across multiple channels via division. It does not constrict bandwidth's digital signals or frequencies. It is less susceptible to interference, thus providing better data communication capability and a more secure private line.

Code Division Multiple Access

When CDM is used to allow multiple signals from multiple users to share a common communication channel, the technology is called Code Division Multiple Access (CDMA). Each group of users is given a shared code and individual conversations are encoded in a digital sequence. Data is available on the shared channel, but only those users associated with a particular code can access the data.

Concept

Each communicating station is assigned a unique code. The codes stations have the following properties –

· If code of one station is multiplied by code of another station, it yields 0. · If code of one station is multiplied by itself, it yields a positive number equal to the number of stations.

The communication technique can be explained by the following example –

Consider that there are four stations w, x, y and z that have been assigned the codes c_w , c_x, $c_y$and c_z and need to transmit data d_w , d_x, d_y and d_z respectively. Each station multiplies its code with its data and the sum of all the terms is transmitted in the communication channel.

Thus, the data in the communication channel is $d_w . c_w + d_x . c_x + d_y . c_y + d_z . c_z$

Suppose that at the receiving end, station z wants to receive data sent by station y. In order to retrieve the data, it will multiply the received data by the code of station y which is d_y.

$$\begin{aligned} data &= (d_w . c_w + d_x . c_x + d_y . c_y + d_z . c_z) . c_y \\ &= d_w . c_w . c_y + d_x . c_x . c_y + d_y . c_y . c_y + d_z . c_z . c_y \\ &= 0 + 0 + d_y . 4 + 0 = 4d_y \end{aligned}$$

Thus, it can be seen that station z has received data from only station y while neglecting the

other codes.

Orthogonal Sequences

The codes assigned to the stations are carefully generated codes called chip sequences or more popularly orthogonal sequences. The sequences are comprised of +1 or –1. They hold certain properties so as to enable communication.

The properties are –

A sequence has m elements, where m is the number of stations.

· If a sequence is multiplied by a number, all elements are multiplied by that number. · For multiplying two sequences, the corresponding positional elements are multiplied and summed to give the result.
· If a sequence is multiplied by itself, the result is m, i.e. the number of stations. ·
If a sequence is multiplied by another sequence, the result is 0.
· For adding two sequences, we add the corresponding positional elements.

Let us ascertain the above properties through an example.

Consider the following chip sequences for the four stations w, x, y and z

– [+1 -1 -1 +1], [+1 +1 -1 -1], [+1 -1 +1 -1] and [+1 +1 +1 +1]

· Each sequence has four elements.
· If [+1 -1 -1 +1] is multiplied by 6, we get [+6 -6 -6 +6].
· If [+1 -1 -1 +1] is multiplied by itself, i.e. [+1 -1 -1 +1]. [+1 -1 -1 +1], we get +1+1+1+1 = 4, which is equal to the number of stations.
· If [+1 -1 -1 +1] is multiplied by [+1 +1 -1 -1], we get +1-1+1-1 = 0 ·
If [+1 -1 -1 +1] is added to [+1 +1 -1 -1], we get [+2 0 -2 0].

The commonly used orthogonal codes are **Walsh codes**.

UNIT-3

Data Link Layer: - The Data Link Layer - Services Provided to the Network Layer – Framing – Error Control – Flow Control, Error Detection and Correction – Error-Correcting Codes – Error Detecting Codes, Elementary Data Link Protocols- A Utopian Simplex Protocol-A Simplex Stop and Wait Protocol for an Error free channel-A Simplex Stop and Wait Protocol for a Noisy Channel, Sliding Window Protocols-A One Bit Sliding Window Protocol-A Protocol Using Go-Back-NA Protocol Using Selective Repeat

DATA LINK LAYER DESIGN ISSUES

DATA LINK LAYER

The data link layer has a number of specific functions it can carry out. These functions include

1. Providing a well-defined service interface to the network layer.
2. Provides services for the reliable interchange of data across a data link established by the physical layer.
3. Link layer protocols manage the establishment, maintenance, and release of data-link connections.
4. Data-link protocols control the flow of data, dealing with transmission errors, and supervise data error recovery.
5. Regulating the flow of data so that slow receivers are not swamped by fast senders.
6. Recovery from abnormal conditions.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into frames for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer, as illustrated in Fig. 2-1. Frame management forms the heart of what the data link layer does.

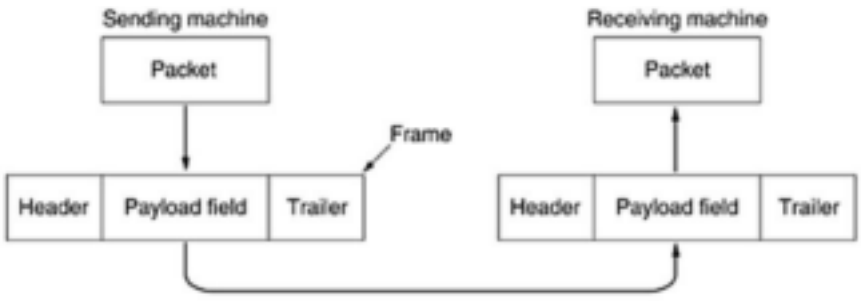


Figure 2-1. Relationship between packets and frames.

Data Link Layer Design Issues

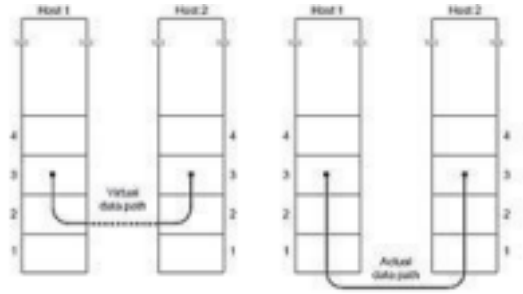
1. Providing a well-defined service interface to the network layer
2. Determining how the bits of the physical layer are grouped into frames
3. Dealing with transmission errors
4. Regulating the flow of frames so that slow receivers are not swamped by fast senders.

1. Services Provided to the Network Layer

▪ The function of the data link layer is to provide service to the network layer. ▪ The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine.

▪ The network layer hands some bits to the data link layer for transmission to the destination, the job of the data link layer is to transmit the bits to the destination machine, so they can be handed over to the network layer on the destination machine.

The job of the data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer there, as shown in Fig.(a). The actual transmission follows the path of Fig.(b), but it is easier to think in terms of two data link layer processes communicating using a data link protocol.



(a) Virtual communication. (b) Actual communication.

▪ The data link layer can be designed to offer various services, Three possibilities that are commonly provided are:

1. Unacknowledged connectionless service.

2. Acknowledged connectionless service.

3. Acknowledged connection-oriented service.

Unacknowledged connectionless service

Unacknowledged connectionless service consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them. No connection is established beforehand or released afterward. Good channels with low eror rates, for real-time traffic, such

as speech.

Acknowledged connectionless service

When this service is offered, there are still no connections used, but each frame sent is individually acknowledged. This way, the sender knows whether or not a frame has arrived safely. Good for unreliable channels, such as wireless.

Acknowledged connection-oriented service

With this service, the source and destination machines establish a connection before any data are transferred. Each frame sent over the connection is numbered, and the data link layer guarantees that each frame sent is received. Furthermore, it guarantees that each frame is received exactly once and that all frames are received in the right order.

When connection-oriented service is used, transfers have three distinct phases.

1. In the first phase the connection is established by having both sides initialize variable and counter need to keep track of which frames have been received and which ones have not.
2. In the second phase, one or more frames are actually transmitted.
3. In the third phase, the connection is released, freeing up the variables, buffers, and other resources used to maintain the connection.

2. Framing

- In order to provide service to the network layer, the data link layer must use the service provided to it by the physical layer.
- What the physical layer does is accept raw bit stream and attempt to deliver it to the destination. This bit stream is not guaranteed to be error free.
- It is up to the data link layer to detect, and if necessary, correct errors.
- The usual approach is for the data link layer to break the bit stream up into discrete frames and compute the checksum for each frame. When the frames arrive at the destination , the checksum is re-computed.

There are four methods of breaking up the bit stream

1. Character count.
2. Starting and ending character stuffing.
3. Starting and ending flags, with bit stuffing.
4. Physical layer coding violations.

Character count, uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count, it knows how many characters follow. Problem: count can possible be misrepresented by a transmission error. This method is rarely used anymore.

Starting and ending character stuffing, gets around the problem of resynchronization after an error by having each frame start with the ASCII character sequence DLE STX and end with the sequence DLE ETX. (DLE is Data Link Escape, STX is Start of Text, and ETX is End of Text). **Problem:** a serious

problem occurs with this method when binary data, such as object programs or floating-point numbers, are being transmitted it is possible that the DLE, STX, and ETX characters can occur, which will interfere with the framing. One way to solve this problem is to have the sender's data link layer insert and DLE character just before each "accidental" DLE and the data link layer on the other machine removes them before it gives the data to the network layer, this is called Character stuffing.

Starting and ending flags with bit stuffing, allows data frames to contain and arbitrary number of bits and allows character codes with an arbitrary number of bits per character. Each frame begins and ends with a special bit pattern, 01111110, called a flag byte. Whenever the sender's data link layer encounters five

consecutive ones in the data, it automatically stuffs a 0 bit into the outgoing bit stream, which is called bit stuffing. The receiving machine destuffs the 0 bit. • The fourth method, Physical coding violations, is only applicable to networks in which the encoding on the physical medium contains some redundancy. For example, some LANs encode 1 bit of data by using 2 physical bits.

3. **Error Control**

- The next problem to deal with is to make sure all frames are eventually delivered to the network layer at the destination, and in proper order.
- The usual way to ensure reliable delivery is to provide the sender with some feedback about what is happening at the other end of the line.
- One complication with this is that the frame may vanish completely, in which case, the receiver will not react at all, since it has no reason to react.
- This possibility is dealt with by introducing timers into the data link layer. When the sender transmits a frame, it generally also starts a timer. The timer is set to go off after an interval long enough for the frame to reach the destination machine. If the frame or acknowledgment is lost the timer will go off. The obvious solution is to transmit the frame again. This creates the problem of possible sending frames multiple times. To prevent this from happening, it is generally necessary to assign sequence numbers to outgoing frames, so that the receiver can distinguish retransmission from originals.
- The whole issue of managing the timers and sequence numbers so as to ensure that each frame is ultimately passed to the network layer at the destination exactly one, no more no less, is an important part of the data link layer's duties.

4. Flow Control

- Another important design issue that occurs in the data link layer (and higher layers as well) is what to do with a sender that systematically wants to transmit frames faster than a receiver can accept them.
- This situation can easily occur when the sender is running on a fast computer and the receiver is running on a slow machine.
- The usual solution is to introduce flow control to throttle the sender into sending no faster than the receiver can handle the traffic.
- Various flow control schemes are known, but most of them use the same basic principle, eg HDLC. The protocol contains well-defined rules about when a sender may transmit the next frame.

ERROR DETECTION AND CORRECTION

Error sources are present when data is transmitted over a medium. Even if all possible error-reducing measures are used during the transmission, an error invariably creeps in and begins to disrupt data transmission. Any computer or communication network must deliver accurate messages.

Error detection is applied mostly in the data-link layer but is also performed in other layers. In some cases, the transport layer includes some sort of error-detection scheme. When a packet arrives at the destination, the destination may extract an error-checking code from the transport header and perform error detection. Sometimes, network-layer protocols apply an error-detection code in the network-layer header. In this case, the error detection is performed only on the IP header, not on the data field. At the application layer, some

type of error check, such as detecting lost packets, may also be possible. But the most common place to have errors is still the data-link layer. Possible and common forms of errors at this level are described here and are shown in Figure 2.1

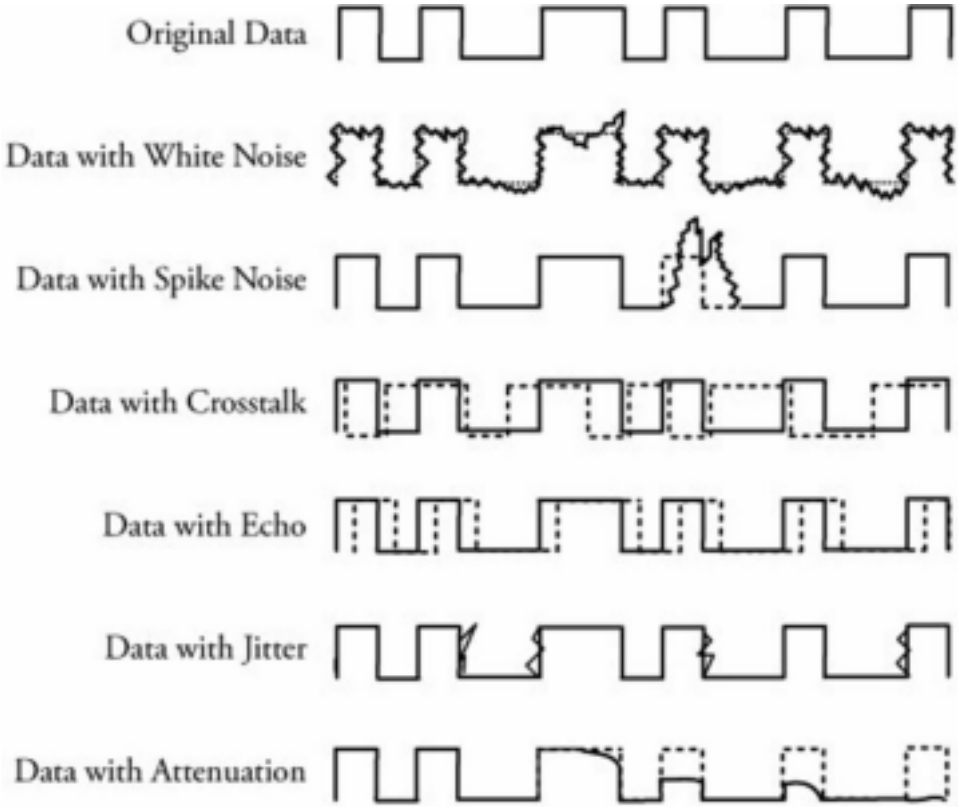


Figure 2.1. Common forms of data errors at the data-link level

Networks must be able to transfer data from one device to another with complete accuracy. Data can be corrupted during transmission. For reliable communication, errors must be detected and corrected.

TYPES OF ERRORS

Whenever bits flow from one point to another, they are subject to unpredictable changes because of interference. This interference can change the shape of the signal; leads to an error in data transmission.

Basic types of errors are

i. Single-Bit Error
ii. Burst Error

i. Single-Bit Error

The term single-bit error means that only one bit of a given data unit (such as a byte, character, data unit, or packet) is changed from 1 to 0 or from 0 to I. In a single-bit error, only one bit in the data unit has changed.

Figure 2.1 shows the effect of a single-bit error on a data unit. To understand the impact of the change, imagine that each group of 8 bits is an ASCII character with a 0 bit added to the left. In the figure, 00000010 (ASCII STX) was sent, meaning start of text, but 00001010 (ASCII LF) was received, meaning line feed.

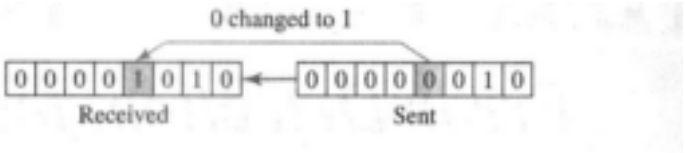


Fig 2.1 Single-bit error

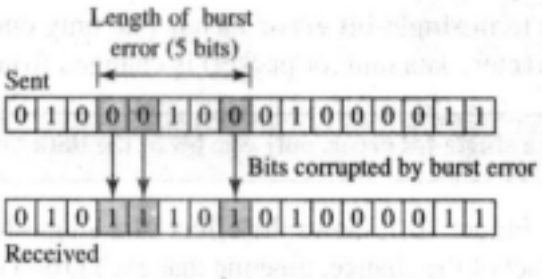
Single-bit errors are the least likely type of error in serial data transmission. To understand Why, imagine a sender sends data at 1 Mbps means that each bit lasts only 1/1,000,000 s, or 1 µs. For a single-bit error to occur, the noise must have a duration of on1y 1 µs, which is very rare; noise normally lasts much longer than this.

However, a single-bit error can happen if we are sending data using parallel transmission. For example, if eight wires are used to send all 8 bits of 1 byte at the same time and one of the wires is noisy, one bit can be corrupted in each byte. Think of parallel transmission inside a computer, between CPU and memory, for example.

ii. Burst Error

The term burst error means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.

Figure below shows the effect of a burst error on a data unit. In this case, 0100010001000011 was sent, but 0101110101000011 was received. Note that a burst error does not necessarily mean that the errors occur in consecutive bits. The length of the burst is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not been corrupted.



Burst error is most likely to occur in a serial transmission. The duration of noise is normally longer than the duration of one bit, which means that when noise affects data, it affects a set of bits. The number of bits affected depends on the data rate and duration of noise. For example, if we are sending data at 1 Kbps, a noise of 1/100 s can affect 10 bits; if we are sending data at 1 Mbps, the same noise can affect 10,000 bits.

ERROR DETECTION

Most networking equipment at the data-link layer inserts some type of error-detection code. When a frame arrives at the next hop in the transmission sequence, the receiving hop extracts the error-detection code and

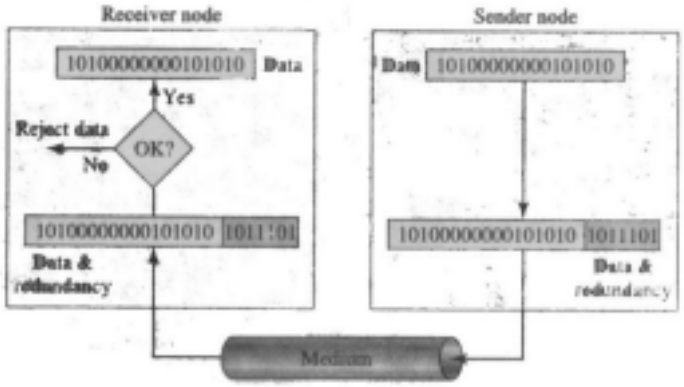
applies it to the frame. When an error is detected, the message is normally discarded. In this case, the sender of the erroneous message is notified, and the message is sent again. However, in real-time applications, it is not possible to resend messages. The most common approaches to error detection are

Redundancy

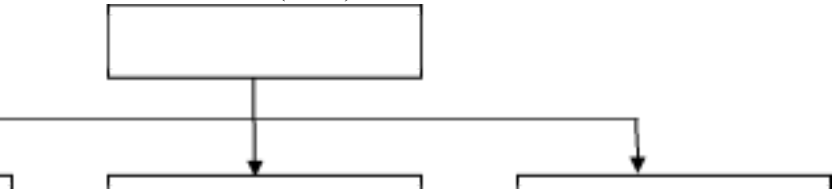
One error detection mechanism would be to send every data unit twice. The receiving device would then able to do a bit for bit comparison between the two versions of the data. Any discrepancy would indicate an error, and an appropriate correction mechanism could be set in place The system would be completely accurate (the odds of errors being introduced on to exactly the same bits in both sets of data are infinitesimally small), but it would also Not only would the transmission time double, but also the time it takes to compare every unit bit by bit must be added.

Error detection uses the concept of redundancy, which means adding extra bits for detecting errors at the destination.. This technique is called redundancy because the extra bits are redundant to the information; they are discarded as soon as the accuracy of the transmission has been determined.

Figure 1.3 shows the process of using redundant bits to check the accuracy of a data unit. Once the data stream has been generated, it passes through a device that analyzes it and adds on an appropriately coded data unit, now enlarged by several bits, travels over the link to the receiver. The receiver puts the entire stream through a checking function, if the received bit stream passes the checking criteria the data portion of the data unit is accepted and the redundant bits are discarded.



Three types of redundancy checks are common in data communications: Parity check, Cyclic Redundancy Check (CRC), and Checksum.

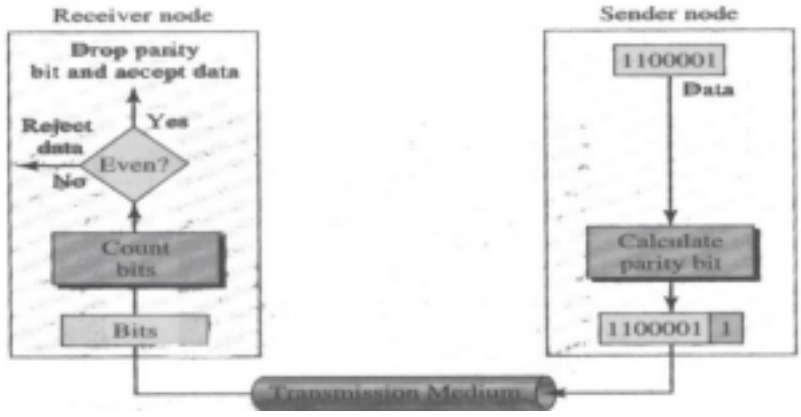


1. PARITY CHECK

The most common and least expensive mechanism for error detection is the parity check. Parity checking can be simple or two dimensional.

Simple Parity Check

In this technique, a redundant bit, called a parity bit, unit so that the total number of Is in the unit (including the parity bit) becomes even (or odd). Suppose we want to transmit the binary data unit 1100001 [ASII a (97)];



Adding the number of 1s gives us 3 an odd number. Before transmitting we pass the data unit through a parity generator. The parity generator counts the 1's and appends the parity bit (a 1 in this case end. The total number of is now 4 an even number. The system now transmits the entire expanded unit across the network link. When it reaches its destination, the receiver puts all 8 bits through an even-parity checking function. If the receiver sees 11000011, it counts four is, an even number, and the data unit passes. But if the data unit has been damaged in transit, means instead of 11000011, the receiver sees 11001011. Then when the parity checker counts the 1s, it gets 5, an odd number. The receiver knows that an error has been introduced into the data somewhere and therefore rejects the whole unit.

Some systems may use odd-parity checking, where the number of Is should be odd. The principle is the same.

Example: Suppose the sender wants to send the word world. In ASCII, the five characters are coded

as 1110111 1101111 1110010 1101100 1100100

w o r l d

Each of the first four characters has an even number of is, so the parity bit is a 0. The last character (d), however, has three 1s (an odd number), so the parity bit is a 1 to make the total number of 1s even. The following shows the actual bits sent (the parity bits are underlined).

1110111<u>0</u> 1101111<u>0</u> 1110010<u>0</u> 1101100<u>0</u> 1100100<u>1</u>

Now suppose the word *world* is received by the receiver without being corrupted in transmission as 1110111<u>0</u> 1101111<u>0</u> 1110010<u>0</u> 1101100<u>0</u> 1100100<u>1</u>, The receiver counts the is in each character and comes up with even numbers (6, 6, 4, 4, 4). The data are accepted.

If the word *world* in is corrupted during transmission and receiver get the data stream

as, 111**1**111<u>0</u> 1101111<u>0</u> 1110**1**10<u>0</u> 1101100<u>0</u> 1100100<u>1</u>

The receiver counts the Is in each character and comes up with even and odd numbers (7, 6, 5, 4, 4). The receiver knows that the data are corrupted, discards them, and asks for retransmission.

Performance

Simple parity check can detect all single bit error It can also detect burst errors as long as the total number of bits change to odd (1, 3, 5, etc.). Let's say we have an even-parity data unit where the total number of 1s, including the parity bit, is 6: 1000111011. If any 3 bits change value, the resulting parity will be odd and the error will be detected: 1**1**1**1**111011:9, 0110111011:7, 1**1**00**0**1**0**011:5 a11 odd. The checker would return a result of 1, and the data unit would be rejected. The same holds true for any odd number of errors.

Suppose, however, that 2 bits of the data unit are changed: 1**11**0111011:8, 1**1**00**0**11011:6, 1000**0**1101**0**:4. In each case the number of is in the data unit is still even. The parity checker will add them and return an even number although the data unit contains two errors. This method cannot detect errors where the total number of bits changed is even. If any two bits change in transmission, the changes cancel each other and the data unit will pass a parity check even though the data unit is damaged. The same holds true for any even number of errors.

Two Dimensional Parity Check

A better approach is the two dimensional parity check. In this method, block of bits is organized as a table (rows and columns). First we calculate the parity bit for each data unit. Then we organize them into a table format.

For example, shown in Figure 1.6, we have four data units shown in four rows and eight columns. We then calculate parity bit for each column and create a new row of 8 bit; they are the parity bits for the whole block. Note that the first parity bit in the fifth row is calculated based on all first bits; the second parity bit is calculated based on all second bits; and so on. We then attach the parity bits to the original data and send them to the receiver.

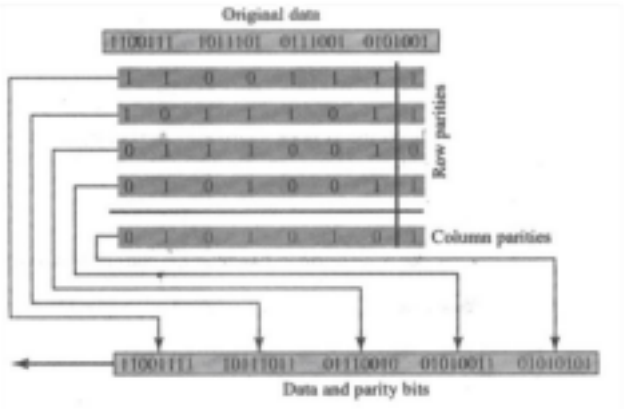


Figure 1.6 Two Dimensional Parity check

Performance

Two-dimensional parity check increases the likelihood of detecting burst errors. A redundancy of *n* bits can easily detect a burst error of n bits. A burst error of more than n bits is also detected by this method with a very high probability. There is, however, one pattern of errors that remains elusive. If 2 bits in one data unit are damaged and two bits in exactly the same positions in another data unit are also damaged, the checker will not detect an error.

2. CYCLIC REDUNDANCY CHECK (CRC)

The most powerful of the redundancy checking techniques is the cyclic redundancy check (CRC). Unlike the parity check which is based on addition, CRC is based on binary division. In CRC, instead of adding bits to achieve a desired parity, a sequence of redundant bits, called the CRC or the CRC remainder, is appended to the end of a data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number. At its destination, the incoming data unit is divided by the same number. If at this step there is no remainder the data unit is assumed to be intact and is therefore accepted. A remainder indicates that the data unit has been damaged in transmit therefore must be rejected.

The redundancy bits used by CRC are derived by dividing the data unit by a pre determined divisor; the remainder is the CRC.

To be valid, a CRC must have two qualities; it must have exactly one less bit than the divisor, and appending it to the end of the data string must make the resulting bit sequence exactly divisible by the divisor.

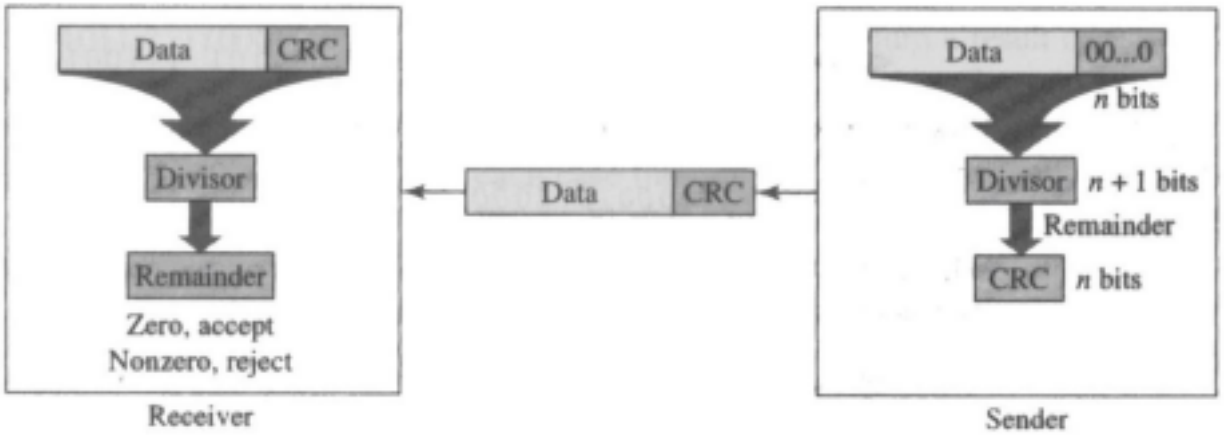


Figure 2.7 CRC generator and checker

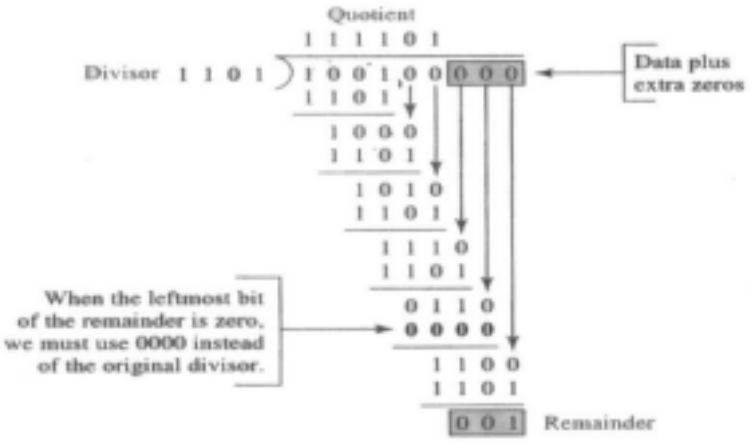
Figure 2.7 provides an outline of the three basic steps in CRC.

1. A string of *n* 0s is appended to the data unit. The number *n* is 1 less than the number of bits in the predetermined divisor, which is *n + 1* bit.
2. The newly elongated data unit is divided by the divisor, using a process called binary division. The remainder resulting from this division is the CRC.
3. The CRC of n bits derived in step 2 replaces the appended 0s at the end of the data unit. Note that the CRC may consist of all 0s.

The data unit arrives at the receiver data first, followed by the CRC. The receiver treats the whole string as a unit and divides it by the same divisor that was used to find the CRC remainder.

If the string arrives without error, the CRC checker yields a remainder of zero and the data unit passes. If the string has been changed in transit, the division yields a nonzero remainder and the data unit does not pass.

The CRC Generator



A CRC generator uses modulo-2 division. Above figure shows this process. In the first step, the 4-bit divisor is subtracted from the first 4 bits of the dividend. Each bit of the divisor is subtracted from the corresponding bit of the dividend without disturbing the next-higher bit. In our example, the divisor, 1101, is subtracted from the first 4 bits of the dividend, 1001, yielding 100 (the leading 0 of the remainder is dropped). The next unused bit from the dividend is then pulled down to make the number of bits in the remainder equal to the number of bits in the divisor. The next step, therefore, is 1000 — 1101, which yields 101, and so on.

In this process, the divisor always begins with a 1; the divisor is subtracted from a portion of the previous that is equal to it in length; the divisor can only be subtracted from a dividend/remainder whose leftmost bit is 1. Anytime the leftmost bit of the dividend/remainder is 0, a string of 0s, of the same length as the divisor, replaces the divisor in that step of the process. For example, if the divisor is 4 bits long, it is replaced by four 0s. (Remember, we are dealing with bit patterns not with quantitative values; 0000 is not the same as 0.) This restriction means that, at any step, the leftmost subtraction will be either 0 - 0 or 1 - 1, both of which equal 0. So, after subtraction, the leftmost bit of the remainder will always be a leading zero, which is dropped, and the next unused bit of the dividend is pulled down to fill out the remainder. Note that only the first bit of the remainder is dropped—if the second bit is also 0, it is retained, and the dividend/remainder for the next step will begin with 0. This process repeats until the entire dividend has been used.

The CRC Checker

A CRC checker functions exactly as the generator does. After receiving the data appended with the CRC, it does the same modulo-2 division. If the remainder is all 0s, the CRC is dropped and the data are accepted; otherwise, the received stream of bits is discarded and data are resent.

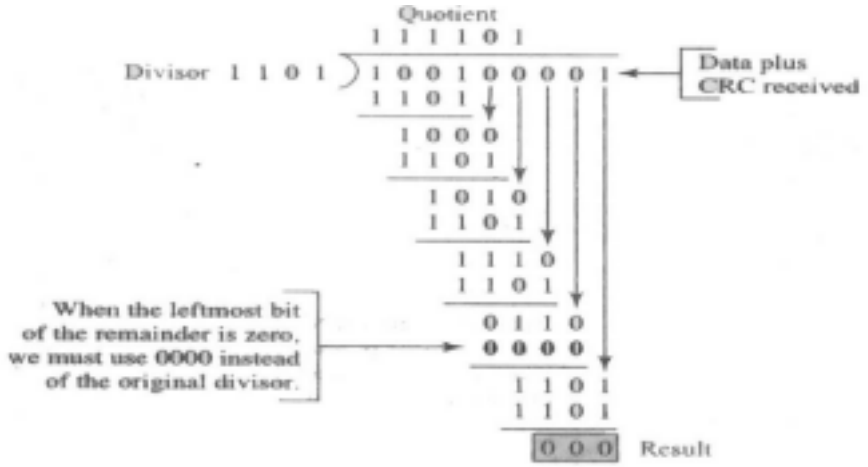


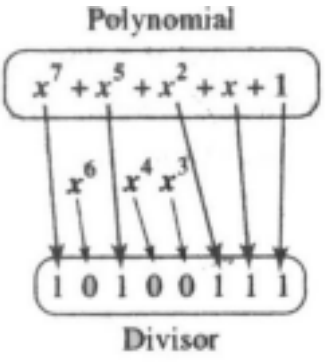
Figure above shows the same process of division in the receiver. We assume that there is no error. The remainder is therefore all 0s, and the data are accepted.

Polynomials

The divisor in the CRC generator is most often represented *not as a string* of 1s and 0s, but as an algebraic polynomial, like $\mathbf{x^7+x^5+x^2+x+1}$

The polynomial format is useful for two reasons: It is short, and it can be used to prove the concept mathematically.

The relationship of a polynomial to its corresponding binary representation is shown below.



A polynomial should be selected to have at least the following properties:

1. It should not be divisible by x.
2. It should be divisible by x + 1.

The first condition guarantees that all burst errors of a length equal to the degree of the polynomial are detected. The second condition guarantees that all burst errors affecting an odd number of bits are detected.

Example: It is obvious that we cannot choose x (binary 10) or x2 + x (binary 110) as the polynomial because both are divisible by x. However, we can choose x + 1 (binary 11) because it is not divisible by x, but is divisible by x + 1. We can also choose x + 1 (binary 101) because it is divisible by x + I (binary division).

Standard Polynomials

Some standard polynomials used by popular protocols for CRC generation are given

below. CRC-8 for ATM: $x^8 + x^2 + x + 1$

CRC-12: $x^{12} + x^{11} + x^3 + x + 1$

CRC-16: $x^{16} + x^{15} + x^2 + 1$

CRC-CCITT: $x^{16} + x^{15} + x^5 + 1$

CRC-32 used in IEEE 802:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

Performance

CRC is a very effective error detection method. If the divisor is chosen according to the previously mentioned rules,

1. CRC can detect all burst errors that affect an odd number of bits.
2. CRC can detect all burst errors of length less than or equal to the degree of the polynomial.
3. CRC can detect, with a very high probability, burst errors of length greater than the degree of the polynomial.

Example The CRC- 12 ($x^{12} + x^{11} + x^3 + x + 1$), which has a degree of 12, will detect all burst errors affecting an odd number of bits, will detect all burst errors with a length less than or equal to 12, and will detect, 99.97 percent of the time, burst errors with a length of 12 or more.

3. CHECKSUM

The third error detection method we discuss here is called the checksum. Like the parity checks and CRC, the checksum is based on the concept of redundancy.

Checksum Generator

In the sender, the checksum generator did the following actions.

1. The unit is subdivided into *k* sections, each of *n* bits.
2. All these *k* sections are added using ones complement arithmetic in such a way that the total is also *n* bits long.
3. The sum is complemented and appended to the end of the original data unit as redundancy bits, called the checksum field
4. The extended data unit is transmitted across the network. So, if the sum of the data segment is T, the checksum will be -T.

Checksum Checker

The receiver follows these steps

1. The unit is divided into k sections, each of n bits as the checksum generation.
2. All sections are added using ones complement to get the sum.
3. The sum is complemented.
4. If the result is zero, the data are accepted: otherwise, they are rejected.

Example: Suppose the following block of 16 bits is to be sent using a checksum of 8 bits. 10101001 00111001

The numbers are added using ones complement arithmetic

10101001
00111001

Sum ~~11100010~~
Checksum 00011101

The pattern sent is: 10101001 00111001 00011101
Data Checksum

Performance

The checksum detects all errors involving an odd number of bits as well as most errors involving an even number of bits. However, if one or more bits of a segment are damaged and the corresponding bit or opposite value in a second segment are also damaged, the sums of those columns will not change and the receiver will not detect a problem. If the last digit of one segment is a 0 and it gets changed to a 1 in transit, then the last 1 in another segment must be changed to a 0 if the error is to go undetected.

In two-dimensional parity check, two 0s could both change to is without altering the parity because carries were discarded. Checksum retains all carries; so although two Os becoming is would not alter the value of their own column, it would change the value of the next-higher column. But anytime a bit inversion is balanced by an opposite bit inversion in the corresponding digit of another data segment, the error is invisible.

FLOW AND ERROR CONTROL

Flow and error control are the main functions of the data link layer. Let us informally define

each. **Flow Control**

Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment. Flow control coordinates the amount of data that can be sent before receiving acknowledgment and is one of the most important duties of the data link layer. The flow of data must not be allowed to overwhelm the receiver.

Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data. The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily. Incoming data must be checked and processed before they can be used. The rate of such processing is often slower than the rate of transmission. For this reason, each receiving device has a block of memory, called a buffer, reserved for storing incoming data until they are processed. If the buffer begins to fill up, the receiver must be able to tell the sender to halt transmission until it is once again able to receive.

Error Control

Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender. In the data link layer, the term error control refers primarily to methods of error detection and retransmission. Error control in the data link layer is often implemented simply: Anytime an error is detected in an exchange, specified frames are retransmitted. This process is called ***automatic repeat request*** (**ARQ**).

FLOW AND ERROR CONTROL MECHANISMS

Three common flow and error control mechanisms are,

1. Stop- and-Wait ARQ,
2. Go-Back-N ARQ
3. Selective-Repeat ARQ.

These are sometimes referred to as ***sliding window protocols***.

1. STOP-AND-WAIT ARQ

Stop-and-Wait ARQ is the simplest flow and error control mechanism. It has the following features:

- The sending device keeps a copy of the last frame transmitted until it receives an acknowledgment for that frame. Keeping a copy allows the sender to retransmit lost or damaged frames until they are received correctly.
- For identification purposes, both data frames and acknowledgment (ACK) frames are numbered alternately 0 and 1. A data 0 frame is acknowledged by an ACK 1 frame, indicating that the receiver

has received data frame 0 and is now expecting data frame 1. This numbering allows for identification of data frames in case of duplicate transmission (important in the case of lost acknowledgment or delayed acknowledgment).

- A damaged or lost frame is treated in the same manner by the receiver. If the receiver detects an error in the received frame, it simply discards the frame and sends no acknowledgment. If the receiver receives a frame that is out of order (0 instead of 1 or 1 instead of 0), it knows that a frame is lost. It discards the out-of-order received frame.
- The sender has a control variable, which we call S, that holds the number of the recently sent frame (0 or 1). The receiver has a control variable, which we call R, that holds the number of the next frame expected (0 or 1).
- The sender starts a timer when it sends a frame. If an acknowledgment is not received within an allotted time period, the sender assumes that the frame was lost or damaged and resends it. ▪ The receiver sends only positive acknowledgment for frames received safe and Sound; it is silent about the frames damaged or lost. The acknowledgment number always defines the number of the next expected frame. If frame 0 is received, ACK I is sent; if frame 1 is received, ACK 0 is sent. **Operation**

In the transmission of a frame, we can have four situations: *normal operation*, the *frame is lost*, *the acknowledgement is lost*, or *the acknowledgment is delayed*.

Normal Operation

In a normal transmission, the sender sends frame 0 and waits to receive ACK 1. When ACK 1 is received, it sends frame 1 and then waits to receive ACK 0, and so on. The ACK must be received before the timer set for each frame expires. Figure below shows successful frame transmissions

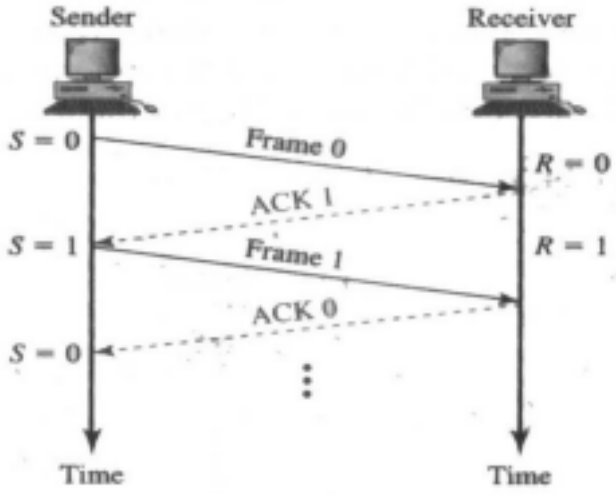


Figure: Stop and Wait ARQ, Normal operation

Lost or Damaged Frame

A lost or damaged frame is handled in the same way by the receiver; when the receiver receives a damaged frame, it discards it, which essentially means the frame is lost. The receiver remains silent about a lost

frame and keeps its value of R. For example, in Figure below, the sender transmits frame 1, but it is lost. The receiver does nothing, retaining the value of R (1). After the timer at the sender site expires, another copy of frame 1 is sent.

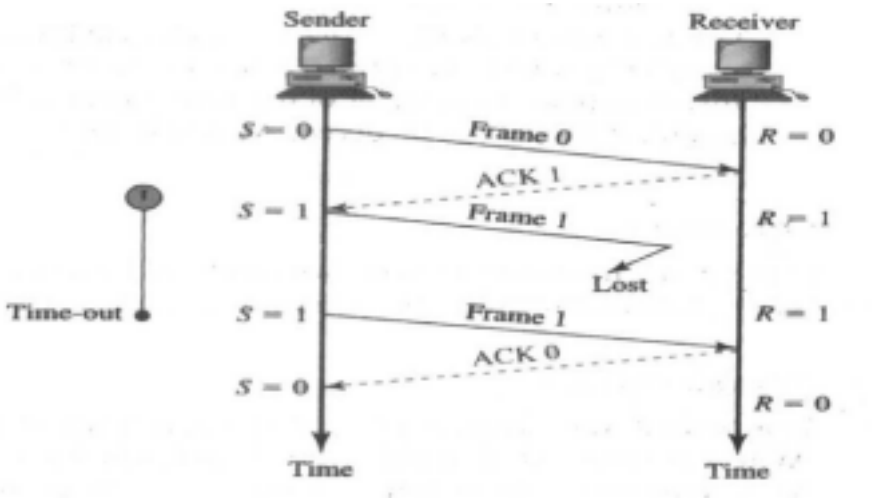


Figure: Stop and Wait ARQ, Lost or Damaged Frame

A lost or damaged acknowledgment

A lost or damaged acknowledgment is handled in the same way by the sender; if the sender receives a damaged acknowledgment, it discards it. Figure below shows a lost ACK 0. The waiting sender does not know if frame I has been received. When the timer for frame 1 expires, the sender retransmits frame 1. Note that the receiver has already received frame 1 and is expecting to receive frame 0 (R = 0). Therefore, it silently discards the second copy of frame 1.

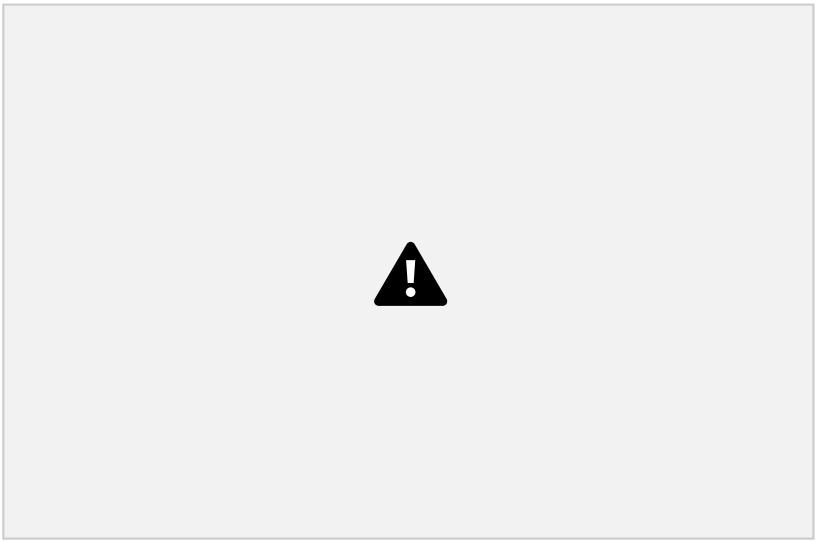


Figure: Stop and Wa1t ARQ, lost ACK frame

Delayed acknowledgment

Another problem that may occur is delayed acknowledgment. An acknowledgment can be delayed at the receiver or by some problem with the link. Figure 4 shows the delay of ACK 1; it is received after the timer for frame 0 has already expired. The sender has already retransmitted a copy of frame 0. However, the value of R at the receiver site is still 1, which means that the receiver expects to see frame 1. The receiver, therefore, discards the duplicate frame 0.

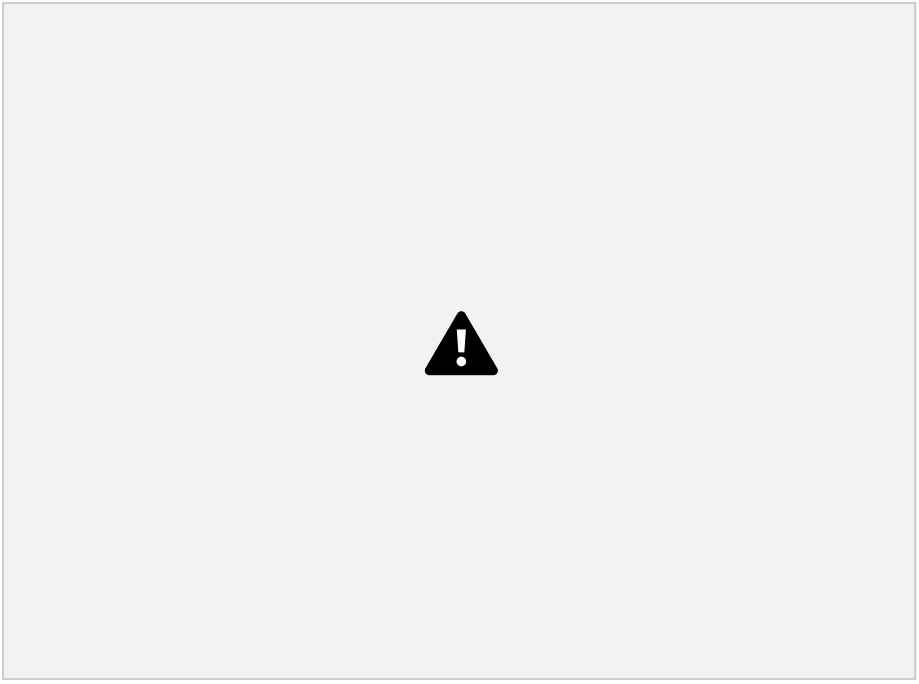


Figure 4: Stop and Wa1t ARQ, delayed ACK frame

The sender has now received two ACKs, one that was delayed and one that was sent after the duplicate frame 0 arrived. The second ACK 1 is discarded.

After the delayed ACK 1 reaches the sender, frame I is sent. However, frame 1 is lost and never reaches the receiver. The sender then receives an ACK 1 for the duplicate frame sent. If the ACKs were not numbered, the sender would interpret the second ACK as the acknowledgment for frame 1. Numbering the ACKs

provides a method to keep track of the received data frames.

Bidirectional Transmission

If the two parties have two separate channels for full- duplex transmission or share the same channel for half-duplex transmission. In this case, each party needs both S and R variables to track frames sent and expected.

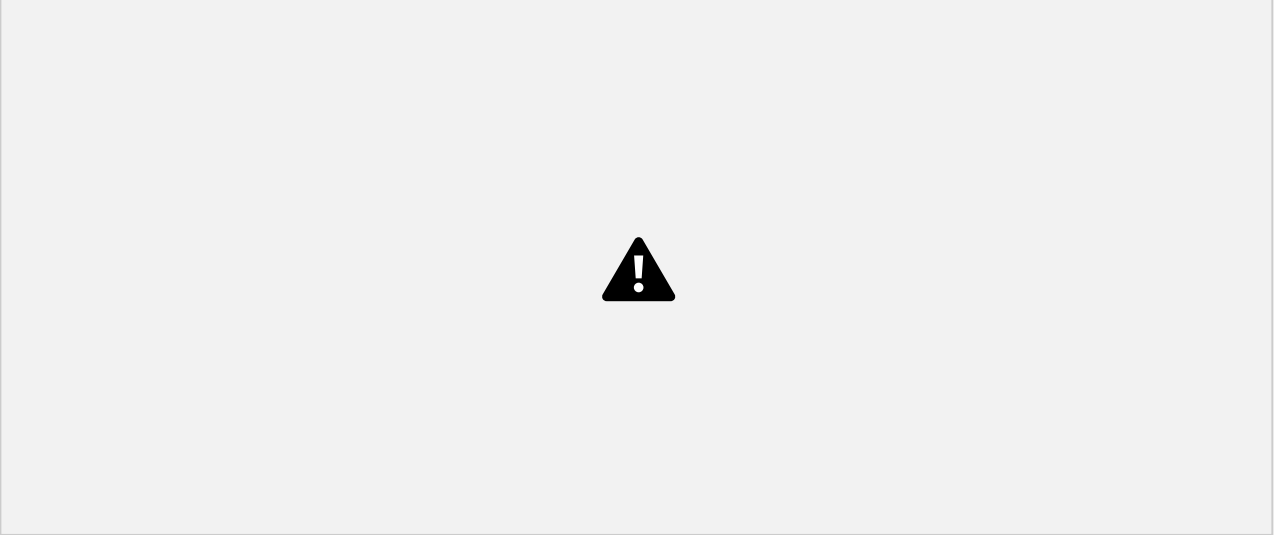


Figure 5: Bidirectional Transmission, Piggybacking

Piggybacking

Piggybacking is a method to combine a data frame with an acknowledgment. For example, in Figure 5, Stations A and B both have data to send. Instead of sending separate data and ACK frames, station A sends a data frame that includes an ACK. Station B behaves in a similar manner.

Piggybacking can save bandwidth because the overhead from a data frame and an ACK frame (addressees, CRC, etc.,) can be combined into just one frame.

2. GO-BACK-N ARQ

In Stop-and-Wait ARQ, at any point in time for a sender, there is only one frame, the outstanding frame,

that is sent and waiting to be acknowledged. This is not a good use of the transmission medium. To improve the efficiency, multiple frames should be in transition while waiting for acknowledgment. In other words, we need to let more than one frame be outstanding. Two protocols use this concept: Go-Back-N ARQ and Selective Repeat ARQ.

In Go-Back-N ARQ, we can send up to W frames before worrying about acknowledgments; we keep a copy of these frames until the acknowledgments arrive. This procedure requires additional features to be added to Stop-and-Wait ARQ.

Sequence Numbers

Frames from a sending station are numbered sequentially. However, because we need to include the sequence number of each frame in the header, we need to set a limit. If the header of the frame allows m bits for the sequence number, the sequence numbers range from 0 to 2^m-1. For example, if m is 3, the only sequence numbers are 0 through 7 inclusive. However, we can repeat the sequence. So the sequence numbers are

0,l,2,3,4,5,6,7, 0,l,2,3,4,5,6,7….1

Sender Sliding Window

At the sender site, to hold the outstanding frames until they are acknowledged, we use the concept of a window. We imagine that all frames are stored in a buffer. The outstanding frames are enclosed in a window. The frames to the left of the window are those that have already been acknowledged and can be purged; those to the right of the Window cannot be sent until the window slides over them. The size of the window is at most 2^m — 1.

The size of the window in this protocol is fixed, although we can have a variable- size window in other protocols such as TCP. The window slides to include new unsent frames when the correct acknowledgments are received. The Window is a sliding window. For example, in Figure 6a, frames 0 through 6 have been sent. In part b, the window slides two frames over because an acknowledgment was received for frames 0 and 1.

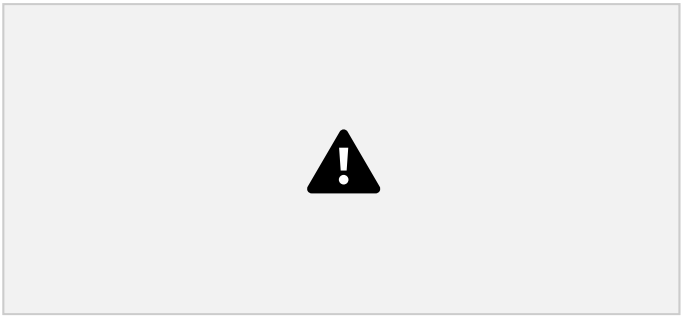


Figure 6: Sender Sliding Window

Receiver Sliding Window

The size of the window at the receiver side in this protocol is always 1. The receiver is always looking for a specific frame to arrive in a specific order. Any frame arriving out of order is discarded and needs to be resent. The receiver window also slides as shown in Figure 7. In part *a* the receiver is waiting for frame 0.

When that arrives, the window slides over.

Figure 6: Sender Sliding Window

Control variables

The sender has three variables, S, S_F, and S_L. The S variable holds the sequence number of the recently sent frame; S_F holds the sequence number of the first frame in the window; and S_L holds the sequence number of the last frame in the window. The size of the window is W, where W= S_L — S_F+ 1.

The receiver only has one variable, R that holds the sequence number of the frame it expects to receive. If the sequence number of the received frame is the same as the value of R, the frame is accepted; if not, it is rejected. Figure 8 shows the sender and receiver window with their control variables.

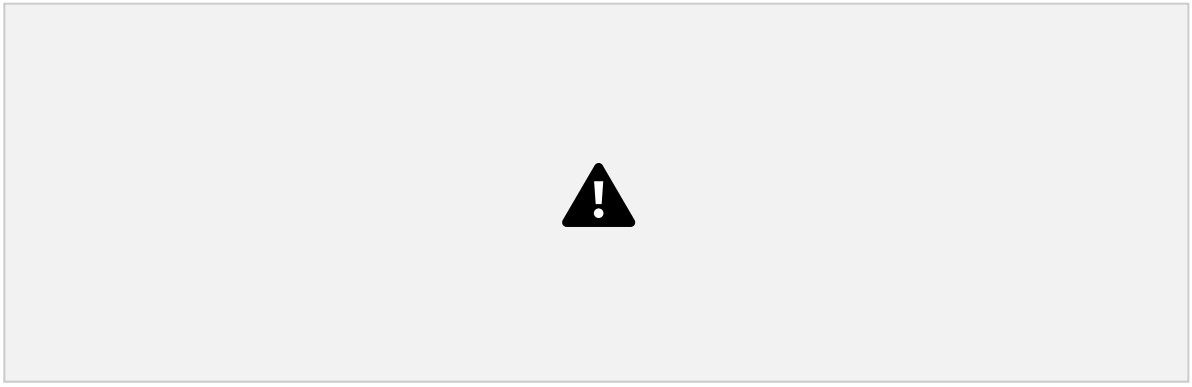


Figure 8, Control variables

Timers

The sender sets a timer for each frame sent. The receiver has no timers.

Acknowledgment

The receiver sends positive acknowledgments if a frame has arrived safe and sound and in order. If a frame is damaged or is received out of order, the receiver is silent and will discard all subsequent frames until it receives the one it is expecting. The silence of the receiver causes the timer of the unacknowledged frame to expire. This, in turn, causes the sender to go back and resend all frames, beginning with the one with the expired timer. The receiver does not have to acknowledge each frame received. It can send one cumulative acknowledgment for several frames.

Resending Frame

When a frame is damaged, the sender goes back and sends a set of frames starting from the damaged one up to the last one sent. For example, suppose the sender has already sent frame 6, but the timer for frame 3 expires. This means that frame 3 has not been acknowledged, so the sender goes back and sends frames 3, 4, 5, 6 again. That is why the protocol is called Go-Back-N ARQ.

OPERATION

Normal Operation

Figure 9 shows a normal operation of Go-Back-N ARQ. The sender keeps track of the outstanding frames and updates the variables and windows as the acknowledgments arrive.

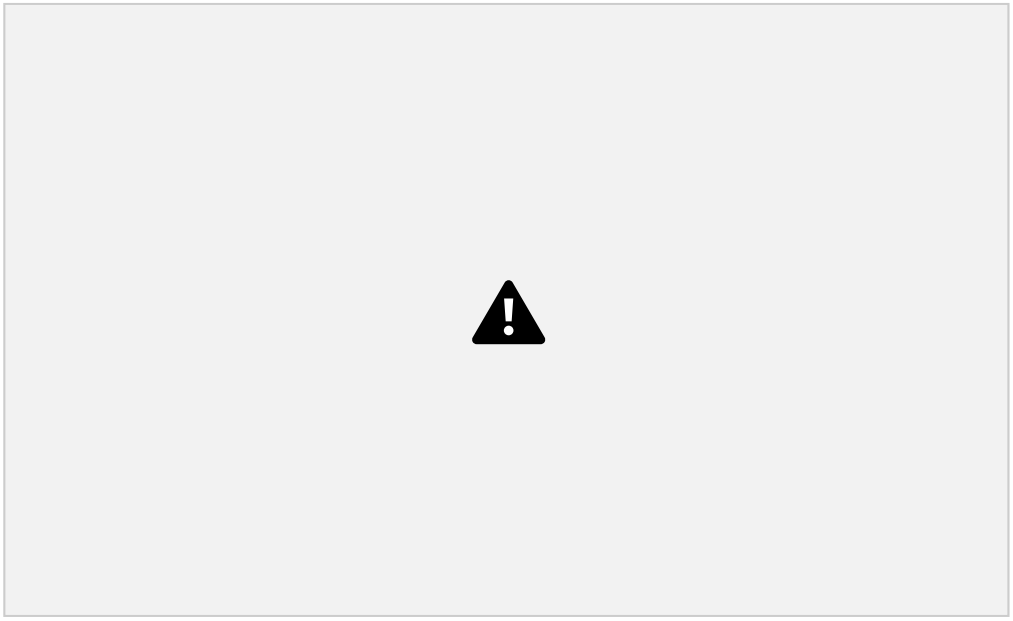


Figure 9 Go-Back-N ARQ, normal operation

Damaged or Lost Frame

Figure 10 shows, the situation when a frame is lost. It shows that frame 2 is lost. Note that when the receiver receives frame 3, it is discarded because the receiver is expecting frame 2, not frame 3 (according to its window). After the timer for frame 2 expires at the sender site, the sender sends frames 2 and 3 (it goes back to 2).

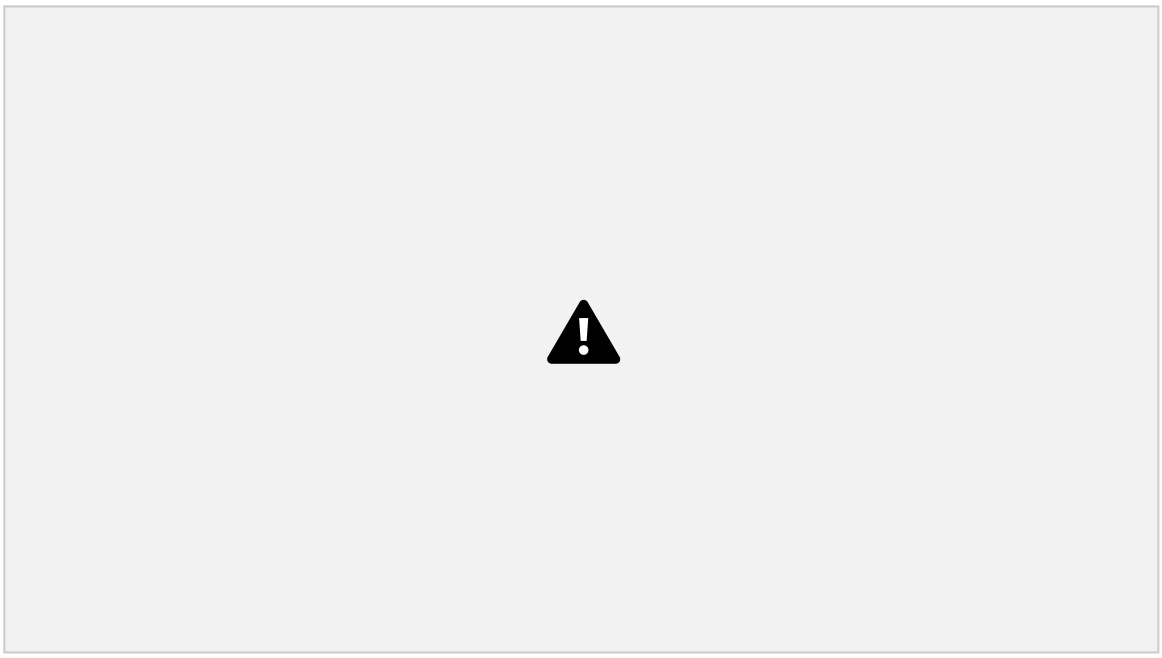


Figure 10: Damaged or Lost Frame

Damaged or lost Acknowledgment

If an acknowledgment is damaged or lost, we can have two situations. If the next acknowledgment arrives before the expiration of any timer, there is no need for retransmission of frames because acknowledgments are cumulative in this protocol. ACK 4 means ACK 1 to ACK 4. So if ACK 1, ACK 2, and ACK 3 are lost, ACK 4 covers them. However, if the next ACK arrives after the time-out, the frame and all the frames after that are resent. The receiver never resends an ACK.

Delayed Acknowledgment

A delayed acknowledgment triggers the resending of frames.

Sender Window Size

We can now show why the size of the sender window must be less than 2^m. As an example, we choose m = 2, which means the size of the window can be 2^m — 1, or 3. Figure 11 compares a window size of 3 and 4.

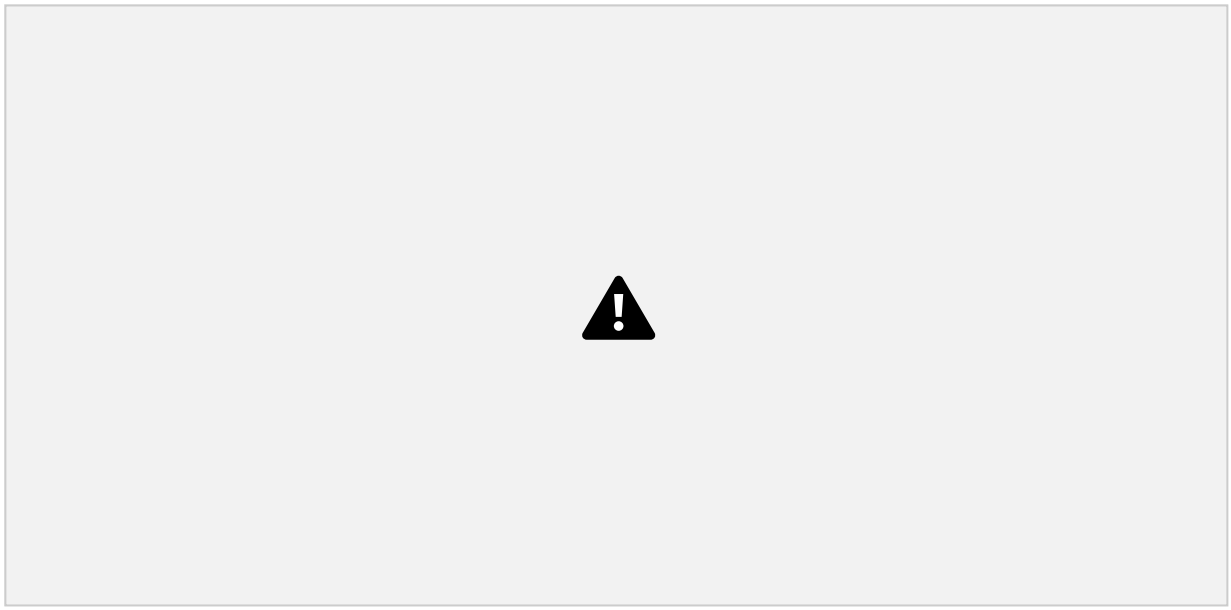


Figure 11 Go-Back-N ARQ: sender Window size

If the size of the window is 3 (less than 22) and all three acknowledgments are lost, the frame 0 timer expires and all three frames are resent. However, the window of the receiver is now expecting frame 3, not frame 0, so the duplicate frame is correctly discarded. On the other hand, if the size of the window is 4 (equal to 22) and all acknowledgments are lost, the sender will send the duplicate of frame 0. However, this time the window of the receiver expects to receive frame 0, so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. This is an error.

Bidirectional Transmission and Piggybacking

As in the case of Stop-and-Wait ARQ, Go-Back N ARQ can also be bidirectional We can also use piggybacking to improve the efficiency of the transmission However, each direction needs both a sender window and a receiver window.

3. SELECTIVE REPEAT ARQ

Go-Back-N ARQ simplifies the process at the receiver site. The receiver keeps3 track of only one variable, and there is no need to buffer out-of-order frames; they are simply discarded. However, this protocol is very inefficient for a noisy link. In a noisy link a frame has a higher probability of damage, which means the resending of multiple frames. This resending uses up the bandwidth and slows down the transmission. For noisy links, there is another mechanism that does not resend N frames when just one frame is damaged; only the damaged frame is resent. This mechanism is called Selective Repeat ARQ. It is more efficient for noisy links, but the processing at the receiver is more complex.

Sender and Receiver Windows

The configuration of the sender and its control variables for Selective Repeat ARQ is the same as those for Go-Back-N ARQ. But the size of the window should be at most one-half of the value $2^{m.}$ The receiver window size must also be this size. This window, however, specifies the range of the accepted received frame. In other words, in Go-Back-N, the receiver is looking for one specific sequence number; in Selective Repeat, the receiver is looking for a range of sequence numbers. The receiver has two control variables R_F and R_L to define the boundaries of the window. Figure 12 shows the sender and receiver windows.

Figure 12 Selective Repeat ARQ, sender and receiver windows

Selective Repeat ARQ also defines a negative acknowledgment (NAK) that reports the sequence number of a damaged frame before the timer expires.

Operation

Figure 13 show the operation of the mechanism of Selective Repeat ARQ with an example of a lost frame.

Frames 0 and I are accepted when received because they are in the range specified by the receiver window. When frame 3 is received, it is also accepted for the same reason. However, the receiver sends a NAK 2 to show that frame 2 has not been received. When the sender receives the NAK 2, it resends only frame 2, which is then accepted because it is in the range of the window.

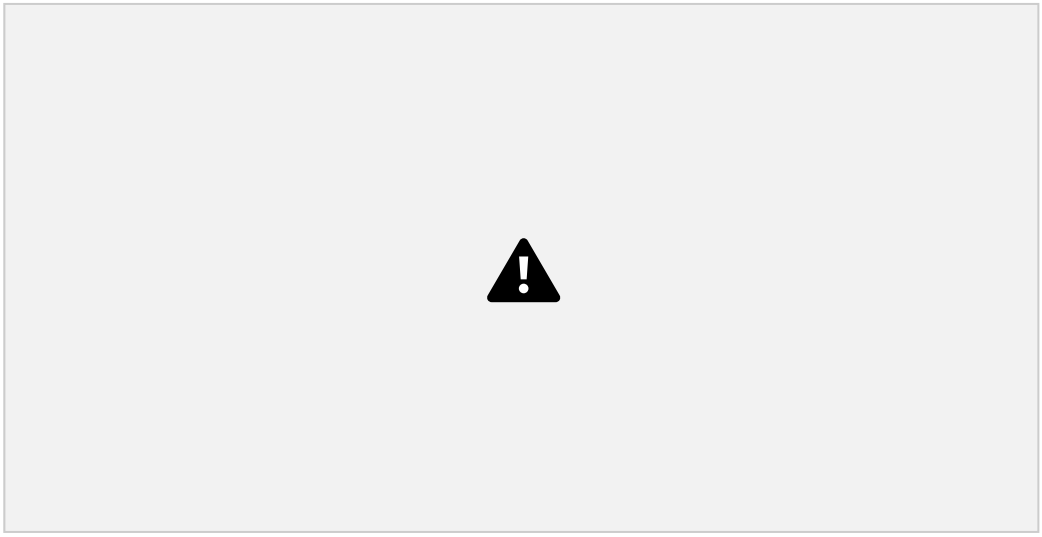


Figure 13 Selective Repeat ARQ, lost frame

Sender Window Size

We can now show why the size of the sender and receiver Windows must be at most one-half of 2^m. For an example, we choose m = 2, which means the size of the Window should be $2^m/2$ or 2. Figure 14 compares a window size of 2 with a window size of 3.

If the size of the window is 2 and all acknowledgments are lost, the timer for frame 0 expires and frame 0 is resent. However, the window of the receiver is now expecting frame 2, not frame 0, so this duplicate frame is correctly discarded. When the size of the window is 3 and all acknowledgments are lost, the sender sends a duplicate of frame 0. However, this time, the window of the receiver expects to receive frame 0 (0 is part of the Window), so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. This is clearly an error.

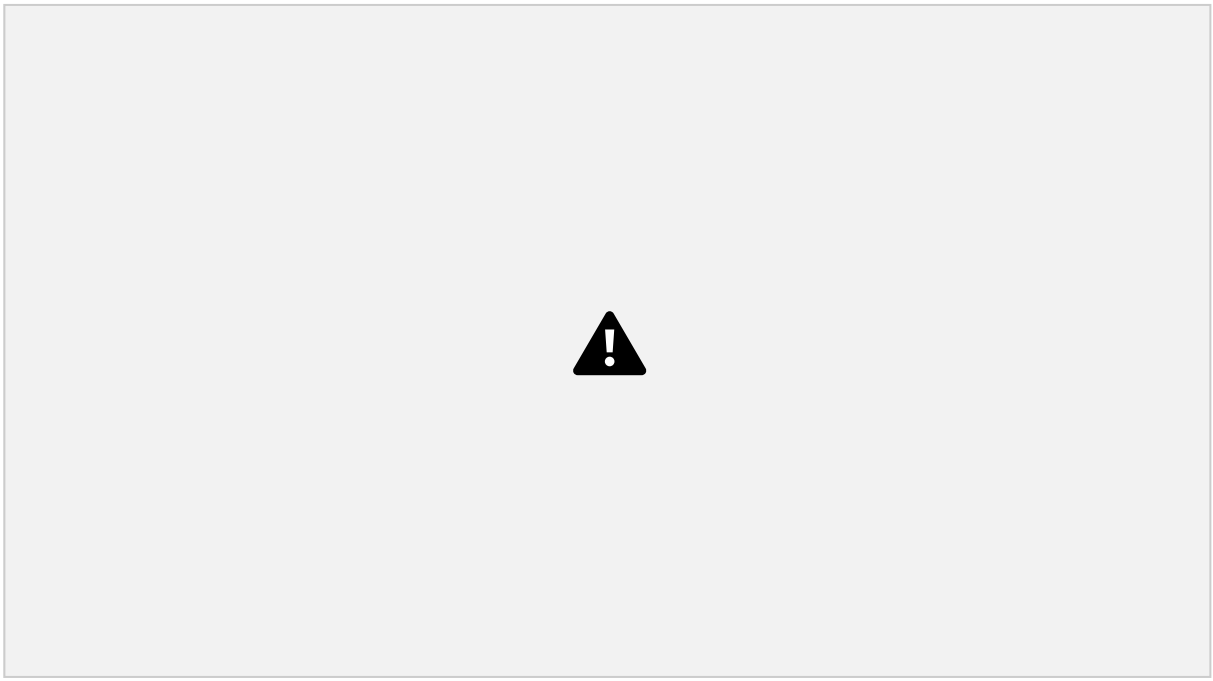


Figure 14 Selective Repeat ARQ, sender window size

Bidirectional Transmission and Piggybacking

As in the case of Stop-and-Wait ARQ and the Go-Back-N ARQ, Selective Repeat ARQ can also be bidirectional. We can use piggybacking to improve the efficiency of the transmission. However each direction needs both a sender window and a receiver window.

UNIT-IV

MEDIUM ACCESS CONTROL SUBLAYER (MAC)

Networks can be categories in to two ways

a) Point to point b) Broad cast channel

- In broadcast network, the key issue is how to share the channel among several users.

- Ex *a conference call with five people*

-Broadcast channels are also called as multi-access channels or random access channels.

-Multi-access channel belong to a sublayer at the DL layer called the MAC sublayer. **The Channel Allocation problem:**

a) **Static channel allocation** in LANs & MANs

i) **FDM** ii) **TDM**

Drawbacks: -1) Channel is wasted if one or more stations do not send data. 2) If users increases this will not support.

b) Dynamic channel allocation

i) Pure **ALOHA** & Slotted **ALOHA**

CSMA/CD

ii) CSMA

CSMA/CA

Pure ALOHA

-1970's Norman Abramson end his colleagues devised this method, used ground –based radio broad costing. This is called the **ALOHA** system.

-The basic idea, many users are competing for the use of a single shared channel. -There are two versions of ALOHA: **Pure and Slotted**.

-Pure ALOHA does not require global time synchronization, where as in slotted ALOHA the time is divided into discrete slots into which all frames must fit.

-Let users transmit whenever they have data to be sent.

-There will be collisions and all collided frames will be damaged.

-Senders will know through feedback property whether the frame is destroyed or not by listening channel.

[-With a LAN it is immediate, with a satellite, it will take 270m sec.]

-If the frame was destroyed, the sender waits random amount of time and again sends the frame.

-The waiting time must be random otherwise the same frame will collide over and

over. USER

A

B

C

D

TIME

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 destroyed.

-We have to find out what is the efficiency of an ALOHA channel?

-Let us consider an infinite collection of interactive users sitting at their systems (stations). -A user will always in two states **typing or waiting**.

-Let the 'Frame time' denotes the time required to transmit one fixed length frame.

-Assume that infinite populations of users are generating new frames according to possion distribution with mean N frames per frame time.

-If N>1 users are generating frames at a higher rate than the channel can handle. -For reasonable throughput 0<N<1.

-In addition to new frames, the station also generates retransmission of frames. -Old and new frames are G per frame time.

-G> N

-At low load there will be few collisions, so G ~ N

-Under all loads, the throughput $S = GP_o$, where P_o is the probability that a frame does not suffer a collision.

-A frame will not suffer a collision if no other frames are sent with one frame time of its start.

-Let 't' be the time required to send a frame.

-If any other user has generated a frame between time t_o and t_o+t, the end of that frame will collide with the beginning of the shaded frame.

-Similarly, any other frame started b/w t_o+t and t_o+2t will bump into the end of the shaded frame.

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

$$P_r[k] = \frac{G^k e^{-G}}{k!}$$

-The probability of zero frames is just e^{-G}

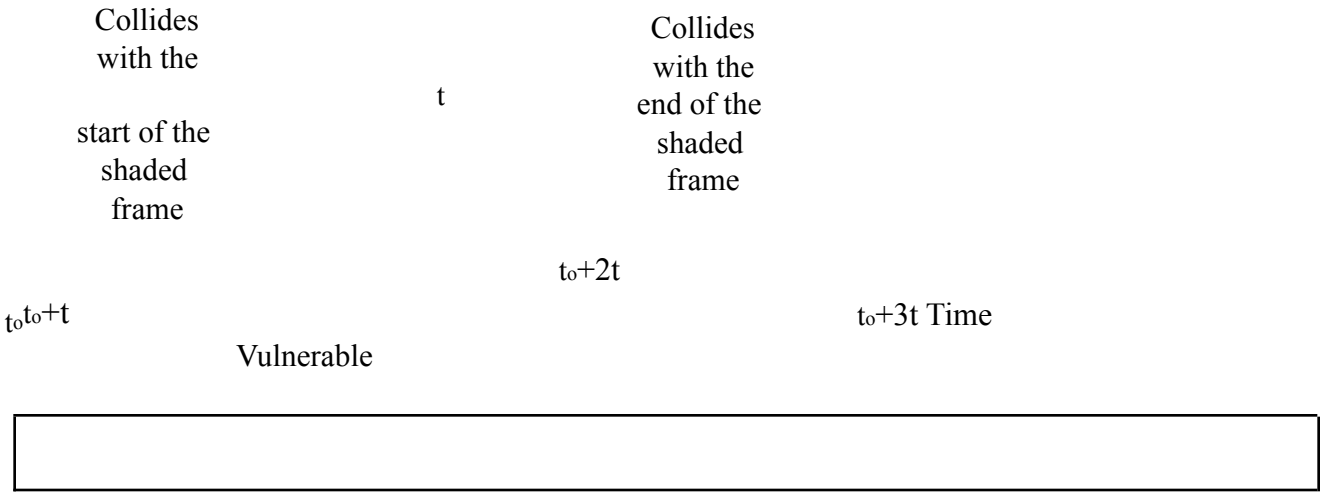
-In an interval two frame times long, the mean number at frames generated is 2G. -The probability at no other traffic being initiated during the entire vulnerable period is given by

$$P_o = e^{-2G} \qquad S = Ge^{-2G}$$

[$S=GP_o$]

The Maximum through put occurs at G=0.5 with S=1/2e = 0.184

The channel utilization at pure ALOHA =18%.



Vulnerable period for the shaded frame

0.368 Slotted ALOHA : S = Ge

-G

0.184

-G

Pure ALOHA : S = Ge

0.5 1.0

G (attempts per packet time)

Throughput versus offered traffic for ALOHA systems

Slotted ALOHA

-In 1972, Roberts' devised a method for doubling the capacity of ALOHA system. -In this system the time is divided into discrete intervals, each interval corresponding to one frame.

-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 of no other traffic during the same slot as our test frame is e^{-G} which leads to

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

- At G=1, slotted ALOHA will have maximum throughput.
- So S=1/e or about 0.368, twice that of pure ALOHA.
- The channel utilization is 37% in slotted ALOHA.

Carrier Sense Multiple Access Protocols

Protocols in which stations listen for a carrier (transmission) and act accordingly are called carries sense protocols.

Persistent CSMA

When a station has data to send, it first listens to the channel to see if any one else is transmitting at that moment. If the channel is busy, the station waits until it become 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 also 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. The longer the propagation delay the worse the performance of the protocol.
Even if the propagation delay is zero, there will be collisions. If two stations listen the channel, that is idle at the same, both will send frame and there will be collision.

Non persistent CSMA

In this, before sending, a station sense the channel. If no one else is sending, the station begins doing so it self. However, if the channel is busy, the station does not continually sense it but it waits a random amount of time and repeats the process.
This algorithms leads to better channel utilization but longer delays then 1-persistent CSMA.

With persistent CSMA, what happens if two stations become active when a third station is busy? Both wait for the active station to finish, then simultaneously launch a packet, resulting a collision. There are two ways to handle this problem.
a) P-persistent CSMA b) exponential backoff.

P-persistent CSMA

The first technique is for a waiting station not to launch a packet immediately when the channel becomes idle, but first toss a coin, and send a packet only if the coin comes up heads. If the coin comes up tails, the station waits for some time (one slot for slotted CSMA), then repeats the process. The idea is that if two stations are both waiting for the medium, this reduces the chance of a collision from 100% to 25%. A simple generalization of the scheme is to use a biased coin, so that the probability of sending a packet when the medium becomes idle is not 0.5, but p, where $0< p < 1$. We call such a scheme **P-persistent CSMA**. The original scheme, where p=1, is thus called 1-persitent CSMA.

Exponential backoff

The key idea is that each station, after transmitting a packet, checks whether the packet transmission was successful. Successful transmission is indicated either by an explicit acknowledgement from the receiver or the absence of a signal from a collision detection circuit. If the transmission is successful, the station is done. Otherwise, the station retransmits the packet, simultaneously realizing that at least one other station is also contending for the medium. To prevent its retransmission from colliding with the other station's retransmission, each station backs off (that is, idles) for a random time chosen from the interval

[0,2*max-propagation_delay] before retransmitting its packet. If the retransmission also fails, then the station backs off for a random time in the interval [0,4* max_propagation_delay], and tries again. Each subsequent collision doubles the backoff interval length, until the retransmission finally succeeds. On a successful transmission, the backoff interval is reset to the initial value. We call this type of backoff exponential backoff.

CSMA/CA

In many wireless LANS, unlike wired LANS, the station has no idea whether the packet collided with another packet or not until it receives an acknowledgement from receiver. In this situation, collisions have a greater effect on performance than with CSMA/CD, where colliding packets can be quickly detected and aborted. Thus, it makes sense to try to avoid collisions, if possible. CSMA/CA is basically p-persistence, with the twist that when the medium becomes idle, a station must wait for a time called the interframe spacing or IFS before contending for a slot. A station gets a higher priority if it is allocated smaller inter frame spacing.

When a station wants to transmit data, it first checks if the medium is busy. If it is, it continuously senses the medium, waiting for it to become idle. When the medium becomes idle, the station first waits for an interframe spacing corresponding to its priority level, then sets a contention timer to a time interval randomly selected in the range [0,CW], where CW is a predefined contention window length. When this timer expires, it transmits a packet and waits for the receiver to send an ack. If no ack is received, the packet is assumed lost to collision, and the source tries again, choosing a contention timer at random from an interval twice as long as the one before(binary exponential backoff). If the station senses that another station has begun transmission while it was waiting for

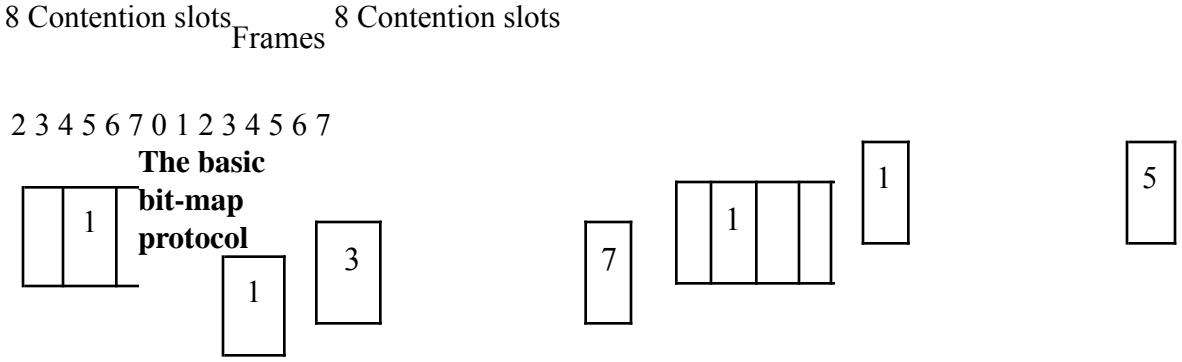
the expiration of the contention timer, it does not reset its timer, but merely freezer it, and restarts the countdown when the packet completes transmission. In this way, stations that happen to choose a longer timer value get higher priority in the next round of contention.

Collision-Free Protocols

A Bit-Map Protocol

In 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 1during slot 1, but only if it has a frame queued. In general, station j may announce the fact 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 with to transmit.



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.

Binary Countdown

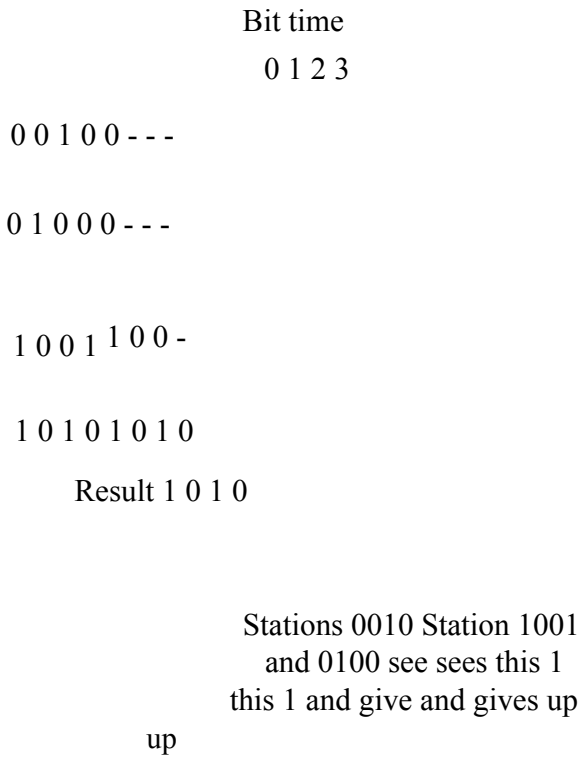
A problem with the basic bit-map protocol is that the overhead is 1 bit per station. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We

will call this protocol binary countdown. It is used in Datakit.

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 station 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. 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 binary countdown protocol. A dash indicates silence



IEEE Standard 802 for LANS and MANS

The IEEE 802.3 is for a 1-persistent CSMA/CD LAN. Xerox built a 2.94 Mbps CSMA/CD system to connect over 100 personal workstations on 1-Km cable. This system was called Ethernet through which electromagnetic radiation was once thought to propagate. Xerox DEC and Intel came with another standard for 100 Mbps Ethernet. This differs from old one that it runs at speeds from 1 to 10 Mbps on various media. The second difference between these two is in one header (802.3 length field is used for packet type in Ethernet).

802.3

Base band Broad band

Digital Analog

(Manchester) (PSK)

10Base5, 10Base2 10 Broad 36 10Base-T, 1Base5

100 Base-T

802.3 Cabling

Five types of cabling are commonly used, 10Base5 cabling called thick Ethernet, came first. It resembles a yellow garden hose, with markings every 2.5 m to show where the taps go. Connections to it are generally made using **vampire taps**, in which a pin is 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 500m.

Name	Cable	Max. segment	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Good for backbones
10Base2	Thin coax	200 m	30	Cheapest system
10Base-T	Twisted pair	100 m	1024	Easy maintenance
10Base-F	Fiber optics	2000 m	1024	Best between buildings

The second cable type was **10Base2** or thin Ethernet, which, in contrast to the garden hose-like thick Ethernet, bends easily. Connections to it are made using industry standard BNC connectors to form T-junctions, rather than using vampire taps. These are

easier to use and more reliable. Thin Ethernet is much cheaper and easier to install, but it can run

for only 200m and can handle only 30 machines per cable segment.

Cable breaks, bad taps, or loose connectors can be detected by a devise called time domain reflectometry.

For 10Base5, a transceiver is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier detection and collision detection. When a collision is detected, the transceiver also puts a

special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

The transceiver cable terminates on an interface board inside the computer. The interface board contains a controller chip that transmits frames to, and receives frames from, the transceiver. The controller is responsible for assembling the data into the proper frame format, as well as computing checksums on outgoing frames and verifying them on incoming frames.

With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver.

With 10Base-T, there is no cable at all, just the hub (a box full of electronics). Adding or removing a station is simple in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100m, may be 150m if high-quality (category 5) twisted pairs are used. 10Base-Tis becoming steadily more popular due to the ease of maintenance. 10Base-F, which uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the method of choice when running between buildings or widely separated hubs.

Each version of 802.3 has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by repeaters. A repeater is a physical layer device. It receives, amplifies, and retransmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different than a single cable (except for some delay introduced by the repeater). A system may contain multiple cable segments and multiple repeaters, but no two transceivers may be more than 2.5km apart and no path between any two transceivers any traverse more than

four repeaters.

A B B

C

Trap

D

Backbone

A
B C D Repeater

802.3 uses Manchester Encoding and differential Manchester Encoding

Bit stream 1 0 0 0 0 1 0 1 1 1 1 Binary encoding

Manchester encoding

Differential Manchester encoding

Transition here Lack of transition here

indicates a 0 indicates a 1

Bytes 7 1 2 or 6 2 or 6 2 0-1500 0-46 4

Preamble		Destination address	Sou addr

Start of frame delimiter

Length of data field

Checksum

The 802.3 MAC sub layer protocol:

I) Preamble:

Each frame start with a preamble of 7 bytes each containing a bit pattern 10101010. **II) Start of frame byte:**

It denotes the start of the frame itself. It contains 10101011.

III) Destination address:

This gives the destination address. The higher order bit is zero for ordinary address and 1for group address (Multi casting). All bits are 1s in the destination field frame will be delivered to all stations (Broad casting).

The 46 bit (adjacent to the high-order bit) is used to distinguish local from global th addresses.

IV) Length field:

This tells how many bytes are present in the data field from 0 to 1500.

V) Data field:

This contains the actual data that the frame contains.

VI) Pad:

Valid frame must have 64 bytes long from destination to checksum. If the frame size less than 64 bytes pad field is used to fill out the frame to the minimum size. **VII) Checksum:**

It is used to find out the receiver frame is correct or not. CRC will be used here.

Other Ethernet Networks

Switched Fast Gigabit

Switched Ethernet:

- 10 Base-T Ethernet is a shared media network.
- The entire media is involved in each transmission.
- The HUB used in this network is a passive device. (not intelligent).
- In switched Ethernet the HUB is replaced with switch. Which is a active device (intelligent)

Fast Ethernet

100 Base_x

100 Base_Tx 100 Base_Fx 100 Base_T4

2 pairs of 2 optical 4 pairs of UTP or STP fibers UTP

Gigabit Ethernet

(optical fiber
fiber 1000
100 Base_Cx STP
Base_Lx 1000
(optical Base_T UTP
100 Base_Sx

multi mode) multi or single
mode)

IEEE 802.4 (Token Bus)

802.3 frames do not have priorities, making them unsuited for real-time systems in which important frames should not be held up waiting for unimportant frames. A simple system with a known worst case is a ring in which the stations take turns sending frames. If there are n stations and it takes T sec to send a frame, no frame will ever have to wait more than nT sec to be sent.

17 20

Broadband

coaxial cable 14 Logical ring

This station not currently in the

13 19 logical ring

11 7

Direction of token motion

This standard, 802.4, describes a LAN called a token bus. Physically, the token bus is a linear or tree-shaped cable onto which the stations are attached. Logically, the stations are organized into a ring, with each station knowing the address of the station to its “left”

and “right.” When the logical ring is initialized, the highest numbered station may send the first frame. After it is done, it passes permission to its immediate neighbor by sending the neighbor a special control frame called a token. The token propagates around the logical ring, with only the token holder being permitted to transmit frames. Since only one station at a time holds the token, collisions do not occur.

Since the cable is inherently a broadcast medium, each station receives each frame, discarding those not addressed to it. When a station passes the token, it sends a token frame specifically addressed to its logical neighbor in the ring, irrespective of where that station is physically located on the cable. It is also worth noting that when stations are first powered on, they will not be in the ring, so the MAC protocol has provisions for adding stations to, and deleting stations from, the ring. For the physical layer, the token bus uses the 75-ohm broadband coaxial cable used for cable television. Both single and dual-cable systems are allowed, with or without head-ends.

Bytes ≥
1 1 2 or 6 2 or 6 0-8182 4 1 1

			Destination address	Source address	Data	Checksum	

Frame control

Start of delimiter End delimiter
Preamble

The frame control field is used to distinguish data frames from control frames. Fro data frames, it carries the frame’s priority. It can also carry an indicator requiring the destination station to acknowledge correct or incorrect receipt of the frame.

For control frames, the frame control field is used to specify the frame type. The allowed types include token passing and various ring maintenance frames, including the mechanism for letting new stations enter the ring, the mechanism for allowing stations to leave the ring, and so on.

Connecting devices

Connecting
devices

Networking Internetworking
devices devices

Repeaters Routers Gateways Connecting devices and the OSI model
Bridges

Application Application Presentation Presentation
Gateway

Session Session Transport Transport

Network Network Data link Data link Physical Physical
Router Bridge Repeate

Bridges

LANS can be connected by devices called bridges, which operate in the data link layer. Bridges do not examine the network layer header and can thus copy IP, IPX, and OSI packets equally well.

The various reasons why the bridges are used.

1) Many university and corporate departments have their own LANS, primarily to connect their own personal computers, workstations, and servers. Since the goals of the various departments differ, different departments choose different LANS, without regard to what other departments are doing. Sooner or later, there is a need for interaction, so bridges are needed.

2) The organization may be geographically spread over several buildings separated by

considerable distances. It may be cheaper to have separate LANS in each building and connect them with bridges and infrared links than to run a single coaxial cable over the entire site.

3) It may be necessary to split what is logically a single LAN into separate LANS to accommodate the load. Putting all the workstations on a single LAN- the total bandwidth needed is far too high. Instead multiple LANS connected by bridges are used. 4) In some situations, a single LAN would be adequate in terms of the load, but the physical distance between the most distant machines is too great (e.g., more than 2.5km for 802.3). Even if laying the cable is easy to do, the network would not work due to the

excessively long round-trip delay. Only solution is to partition the LAN and install bridges between the segments.

5) There is the matter of reliability. On a single LAN, a defective node that keeps outputting a continuous stream of garbage will cripple the LAN. Bridges can be inserted at critical places, to prevent a single node which has gone berserk from bringing down the entire system.

6) And last, bridges can contribute to the organization's security. By inserting bridges at various places and being careful not to forward sensitive traffic, it is possible to isolate parts of the network so that its traffic cannot escape and fall into the wrong hands.

1. Explain the design issues of the Network layer. /

Discuss the services provided to the transport layer by the Network layer. /

How Connection Oriented and Connection Less Services are implemented? Explain. /

Compare the virtual circuits and datagram within the subnet.

NETWORK LAYER DESIGN ISSUES

In the following sections we will provide an introduction to some of the issues that the designers of the network layer must grapple with. These issues include the service provided to the transport layer and the internal design of the subnet.

STORE-AND-FORWARD PACKET SWITCHING

- The network layer protocols operation can be seen in Fig. 5-1.
- The major components of the system are the carrier's equipment (routers connected by transmission lines), shown inside the shaded oval.
- The customers' equipment, shown outside the oval. Host *H1* is directly connected to one of the carrier's routers, *A*, by a leased line. In contrast, *H2* is on a LAN with a router, *F*, owned and operated by the customer. This router also has a leased line to the carrier's equipment.
- We have shown *F* as being outside the oval because it does not belong to the carrier, but in terms of construction, software, and protocols, it is probably no different from the

  carrier's routers. Whether it belongs to the subnet is arguable, but for the purposes of this chapter, routers on customer premises are considered part of the subnet.

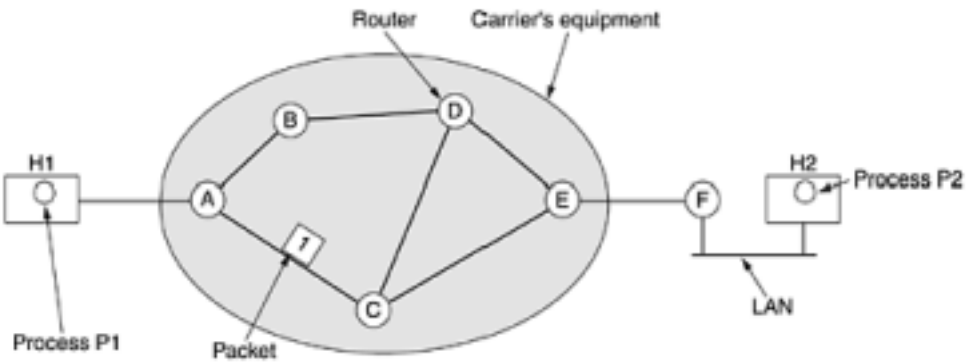


Figure 5-1. The environment of the network layer protocols.

- A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the carrier.

- The packet is stored there until it has fully arrived so the checksum can be verified. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered.
- This mechanism is **store-and-forward packet switching**.

SERVICES PROVIDED TO THE TRANSPORT LAYER

The network layer provides services to the transport layer at the network layer/transport layer interface. The network layer services have been designed with the following goals in mind.

1. The services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

- There is a discussion centers on whether the network layer should provide connection oriented service or connectionless service.
- In their view (based on 30 years of actual experience with a real, working computer network), the subnet is inherently unreliable, no matter how it is designed. Therefore, the hosts should accept the fact that the network is unreliable and do error control (i.e., error detection and correction) and flow control themselves.
- This viewpoint leads quickly to the conclusion that the network service should be connectionless, with primitives SEND PACKET and RECEIVE PACKET and little else.
- In particular, no packet ordering and flow control should be done, because the hosts are going to do that anyway, and there is usually little to be gained by doing it twice.
- Furthermore, each packet must carry the full destination address, because each packet sent is carried independently of its predecessors, if any.
- The other camp (represented by the telephone companies) argues that the subnet should provide a reliable, connection-oriented service.
- These two camps are best exemplified by the Internet and ATM. The Internet offers connectionless network-layer service; ATM networks offer connection-oriented network layer service. However, it is interesting to note that as quality-of-service guarantees are becoming more and more important, the Internet is evolving. In particular, it is starting to acquire properties normally associated with connection-oriented service, as we will see later. Actually, we got an inkling of this evolution during our study of VLANs in Chap. 4.

IMPLEMENTATION OF CONNECTIONLESS SERVICE

- Two different organizations are possible, depending on the type of service offered.
- If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed. In this context, the

  packets are frequently called **datagrams** (in analogy with telegrams) and the subnet is called a **datagram subnet**.

- If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent. This connection is called a **VC** (**virtual circuit**), in analogy with the physical circuits set up by the telephone system, and the subnet is called a **virtual-circuit subnet**.
- Let us now see how a datagram subnet works. Suppose that the process *P1* in **Fig. 5-2** has a long message for *P2*. It hands the message to the transport layer with instructions to deliver it to process *P2* on host *H2*. The transport layer code runs on *H1*, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer, probably just another procedure within the operating system.

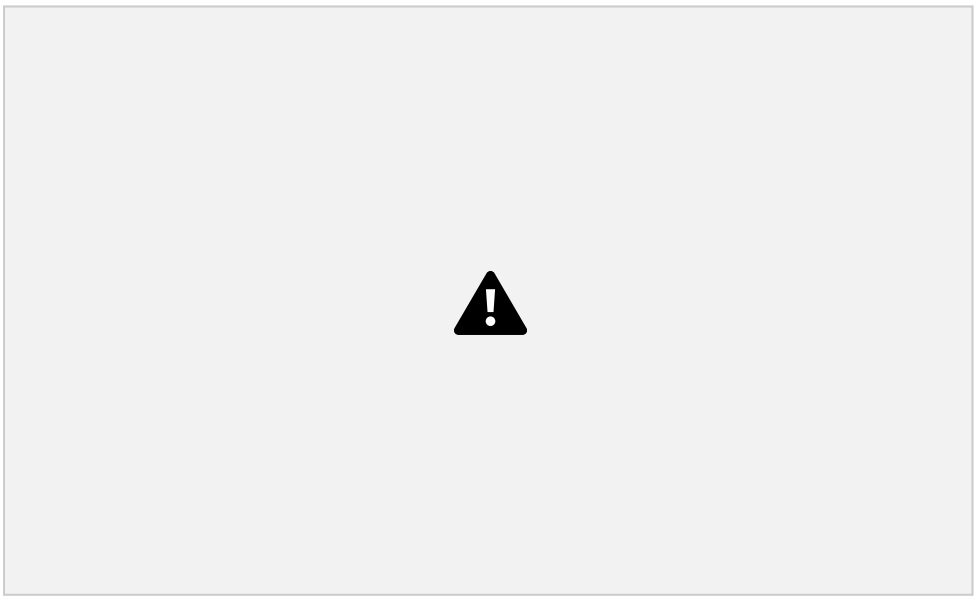


Figure 5-2. Routing within a datagram subnet

- Let us assume that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router *A* using some point-to-point protocol,

For example, PPP. At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination. Only directly connected lines can be used.

For example, in Fig. 5-2, *A* has only two outgoing lines—to *B* and *C*—so every incoming packet must be sent to one of these routers, even if the ultimate destination is some other router. *A*'s initial routing table is shown in the figure under the label "initially". As they arrived at *A*, packets 1, 2, and 3 were stored briefly (to verify their checksums). Then each was forwarded to *C* according to *A*'s table. Packet 1 was then forwarded to *E* and then to *F*. When it got to *F*, it was encapsulated in a data link layer frame and sent to *H2* over the LAN. Packets 2 and 3 follow the same route.

However, something different happened to packet 4. When it got to *A* it was sent to router *B*, even though it is also destined for *F*. For some reason, *A* decided to send packet 4 via a different route than that of the first three. Perhaps it learned of a traffic jam somewhere along the *ACE* path and updated its routing table, as shown under the label "later."

The algorithm that manages the tables and makes the routing decisions is called the **routing algorithm**.

IMPLEMENTATION OF CONNECTION-ORIENTED SERVICE

For connection-oriented service, we need a virtual-circuit subnet.

Let us see how that works.

The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Fig. 5-2. Instead, when a connection is established, a route from the source

machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers.

That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

When the connection is released, the virtual circuit is also terminated. With connection oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

As an example, consider the situation of **Fig. 5-3**. Here, host *H1* has established connection 1 with host *H2*. It is remembered as the first entry in each of the routing tables. The first line of *A*'s table says that if a packet bearing connection identifier 1 comes in from *H1*, it is to be sent to router *C* and given connection identifier 1. Similarly, the first entry at *C* routes the packet to *E*, also with connection identifier 1.

Figure 5-3. Routing within a virtual-circuit subnet.

Now let us consider what happens if *H3* also wants to establish a connection to *H2*. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the subnet to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although *A* can easily distinguish connection 1 packets from *H1* from connection 1 packets from *H3*, *C* cannot do this. For this reason, *A* assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets. In some contexts, this is called label switching.

COMPARISON OF VIRTUAL-CIRCUIT AND DATAGRAM SUBNETS

The major issues are listed in Fig. 5-4, although purists could probably find a counter example for everything in the figure.

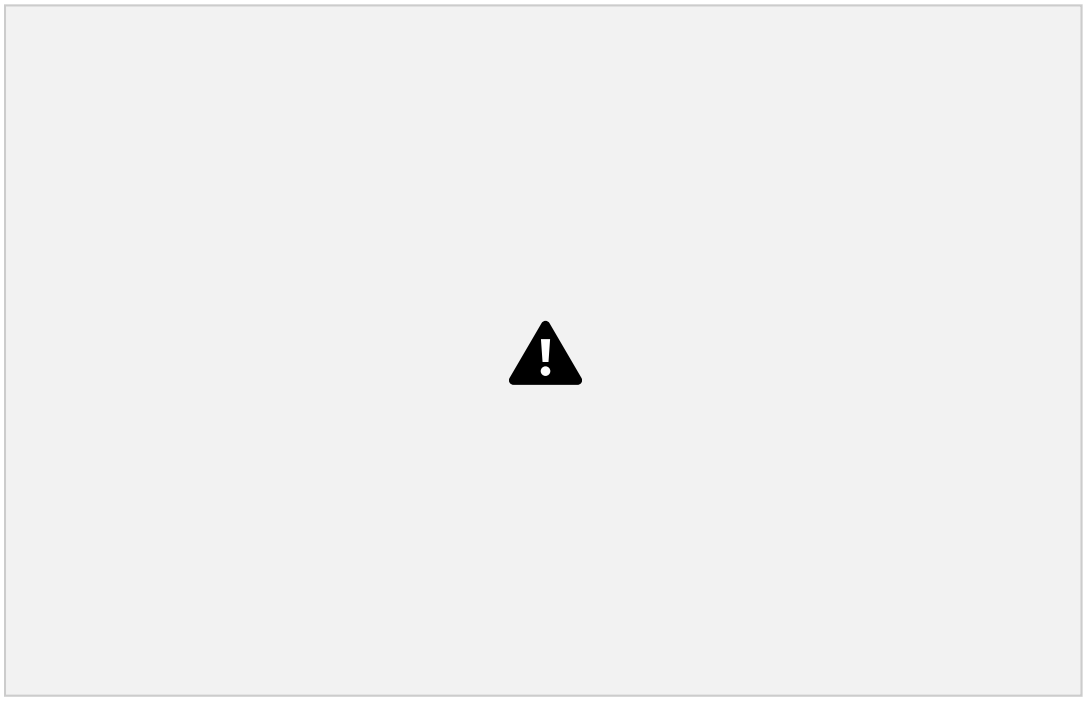


Figure 5-4. Comparison of datagram and virtual-circuit subnets.

2. Discuss about different routing algorithms in detail. (or)

Discuss shortest path routing. (Or)

What is flooding? Discuss. (Or)

Differentiate and explain adaptive and nonadaptive routing algorithms.

(Or) Describe hierarchical Broadcast and Multicasting routing.

(Nov'11, May'10, Dec'08, Nov'07, Dec'05, Dec'04)

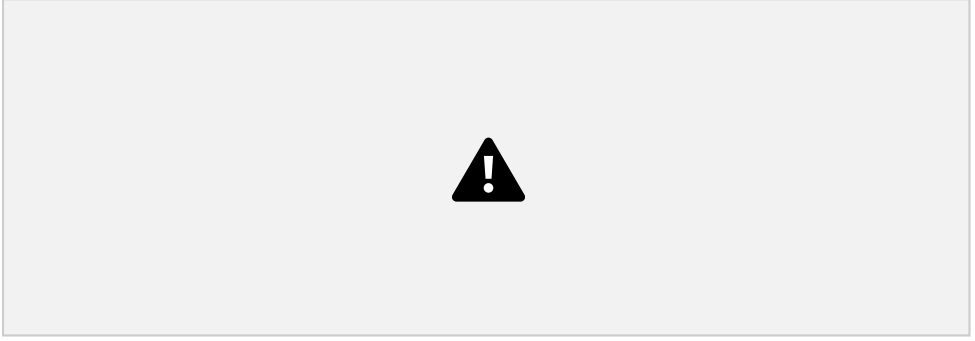
ROUTING ALGORITHMS

The **routing algorithm** is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.

PROPERTIES OF ROUTING ALGORITHM:

Correctness, simplicity, robustness, stability, fairness, and optimality

FAIRNESS AND OPTIMALITY.



Fairness and optimality may sound obvious, but as it turns out, they are often contradictory goals. There is enough traffic between A and A', between B and B', and between C and C' to saturate the horizontal links. To maximize the total flow, the X to

X' traffic should be shut off altogether. Unfortunately, X and X' may not see it that way. Evidently, some compromise between global efficiency and fairness to individual connections is needed.

CATEGORY OF ALGORITHM

Routing algorithms can be grouped into two major classes: **nonadaptive and adaptive.** **Nonadaptive algorithms** do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J is computed in advance, off-line, and downloaded to the routers when the network is booted.

This procedure is sometimes called **Static routing**.

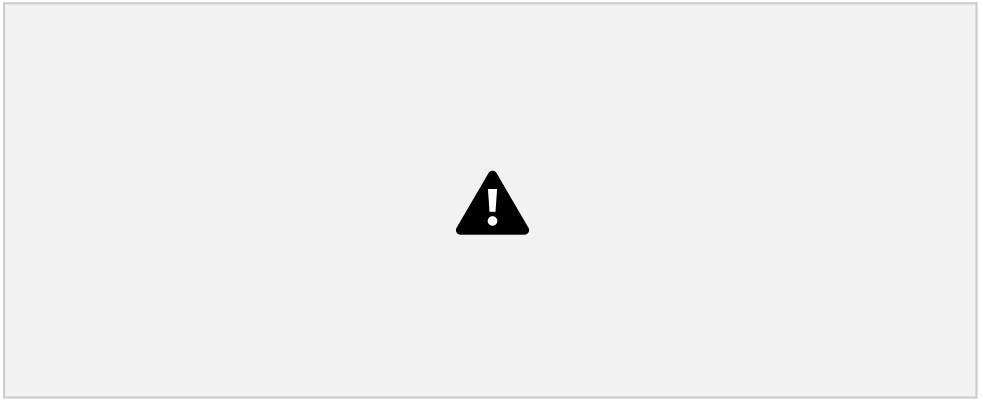
Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well

This procedure is sometimes called **dynamic routing**

THE OPTIMALITY PRINCIPLE

If router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

The set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree.



***(a) A subnet. (b) A sink tree for router* B.**

As a direct consequence of the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a **sink tree** where the distance metric is the number of hops. Note that a sink tree is not necessarily unique; other trees with the same path

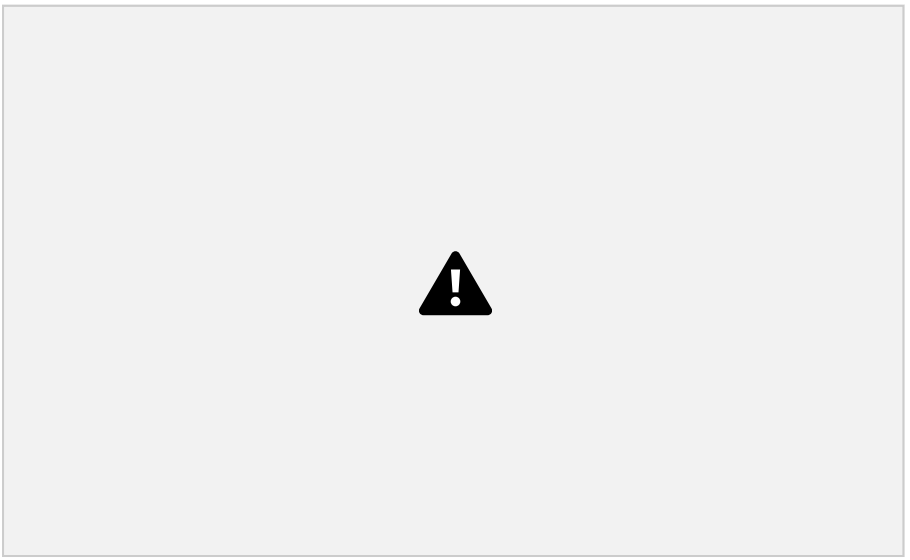
lengths may exist.

The goal of all routing algorithms is to discover and use the sink trees for all routers.

SHORTEST PATH ROUTING

- A technique to study routing algorithms: The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (often called a link).
- To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.
- One way of measuring path length is the number of hops. Another metric is the geographic distance in kilometers. Many other metrics are also possible. For example, each arc could be labeled with the mean queuing and transmission delay for some standard test packet as determined by hourly test runs.
- In the general case, the labels on the arcs could be computed as a function of the distance, bandwidth, average traffic, communication cost, mean queue length, measured

delay, and other factors. By changing the weighting function, the algorithm would then compute the "shortest" path measured according to any one of a number of criteria or to a combination of criteria.



The first five steps used in computing the shortest path from* A *to* D*. The arrows indicate the working node.

To illustrate how the labelling algorithm works, look at the weighted, undirected graph of Fig. 5-7(a), where the weights represent, for example, distance.

We want to find the shortest path from *A* to *D*. We start out by marking node *A* as permanent, indicated by a filled-in circle.

Then we examine, in turn, each of the nodes adjacent to *A* (the working node), relabeling each one with the distance to *A*.

- Whenever a node is relabelled, we also label it with the node from which the probe was made so that we can reconstruct the final path later.
- Having examined each of the nodes adjacent to *A*, we examine all the tentatively labelled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig. 5-7(b).
- This one becomes the new working node.

We now start at *B* and examine all nodes adjacent to it. If the sum of the label on *B* and the distance from *B* to the node being considered is less than the label on that node, we have a shorter path, so the node is relabelled.

After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively-labelled node with the smallest value. This node is made permanent and becomes the working node for the next round. Figure 5-7 shows the first five steps of the algorithm.

- To see why the algorithm works, look at Fig. 5-7(c). At that point we have just made *E* permanent. Suppose that there were a shorter path than *ABE*, say *AXYZE*. There are two possibilities: either node *Z* has already been made permanent, or it has not been. If it has,

then *E* has already been probed (on the round following the one when *Z* was made permanent), so the *AXYZE* path has not escaped our attention and thus cannot be a shorter path.

- Now consider the case where *Z* is still tentatively labelled. Either the label at *Z* is greater than or equal to that at *E*, in which case *AXYZE* cannot be a shorter path than *ABE*, or it is less than that of *E*, in which case *Z* and not *E* will become permanent first, allowing *E* to be probed from *Z*.
- This algorithm is given in Fig. 5-8. The global variables *n* and *dist* describe the graph and are initialized before *shortest path* is called. The only difference between the program and the algorithm described above is that in Fig. 5-8, we compute the shortest path starting at the terminal node, *t*, rather than at the source node, *s*. Since the shortest path from *t* to *s* in an undirected graph is the same as the shortest path from *s* to *t*, it does not matter at which end we begin (unless there are several shortest paths, in which case reversing the search might discover a different one). The reason for searching backward is that each node is labelled with its predecessor rather than its successor. When the final path is copied into the output variable, *path*, the path is thus reversed. By reversing the search, the two effects cancel, and the answer is produced in the correct order.

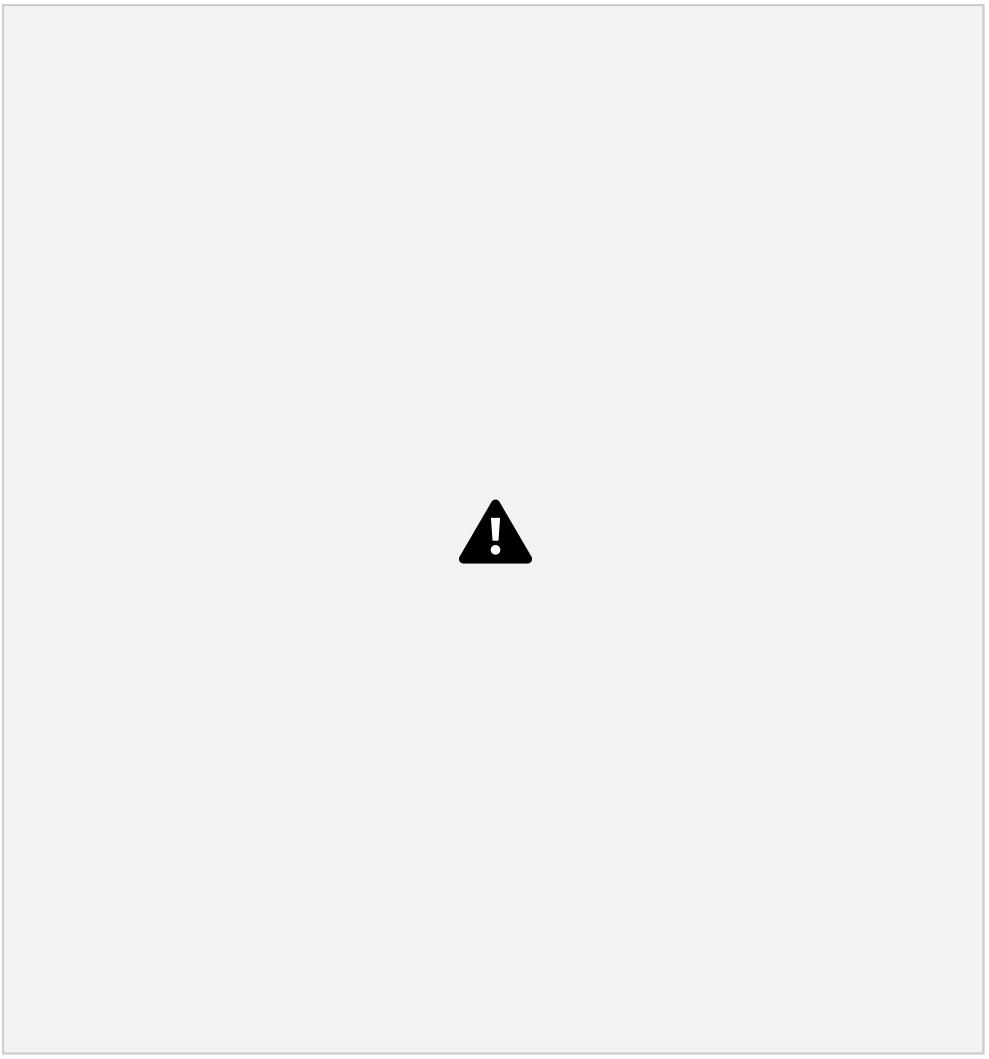


Figure 5-8. Dijkstra's algorithm to compute the shortest path through a graph.

FLOODING

Another static algorithm is **flooding**, in which every incoming packet is sent out on every outgoing line except the one it arrived on.

Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.

One such measure is to have a hop counter contained in the header of each packet, which is decremented at each hop, with the packet being discarded when the counter reaches zero.

Ideally, the hop counter should be initialized to the length of the path from source to destination. If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the subnet.

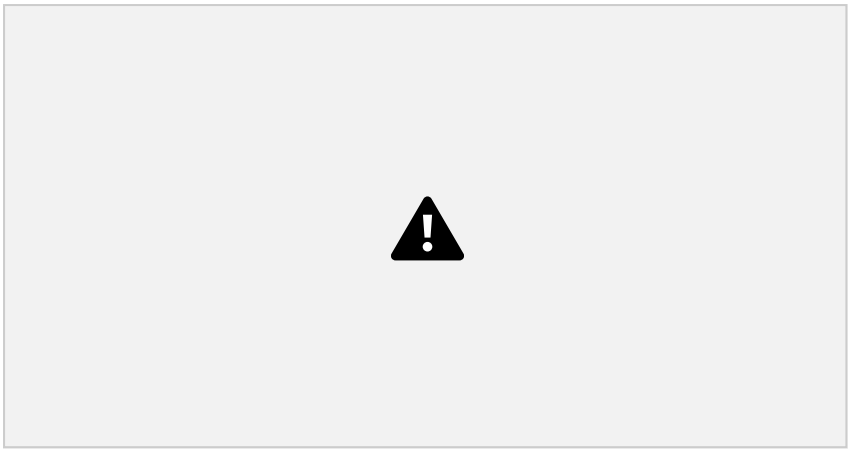
DISTANCE VECTOR ROUTING

Distance vector routing algorithms operate by having each router maintain a table

(i.e, a vector) giving the best known distance to each destination and which line to use to get there.

- These tables are updated by exchanging information with the neighbors.
- The distance vector routing algorithm is sometimes called by other names, most commonly the distributed **Bellman-Ford** routing algorithm and the **Ford-Fulkerson**

  algorithm, after the researchers who developed it (Bellman, 1957; and Ford and Fulkerson, 1962).
- It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.



***(a) A subnet. (b) Input from* A, I, H, K, *and the new routing table for* J.**

- Part (a) shows a subnet. The first four columns of part (b) show the delay vectors received from the neighbours of router *J*.
- *A* claims to have a 12-msec delay to *B*, a 25-msec delay to *C*, a 40-msec delay to *D*, etc. Suppose that *J* has measured or estimated its delay to its neighbours, *A*, *I, H*, and *K* as 8, 10, 12, and 6 msec, respectively.

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

1. The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.
2. A link that is down is assigned an infinite cost.

Example.

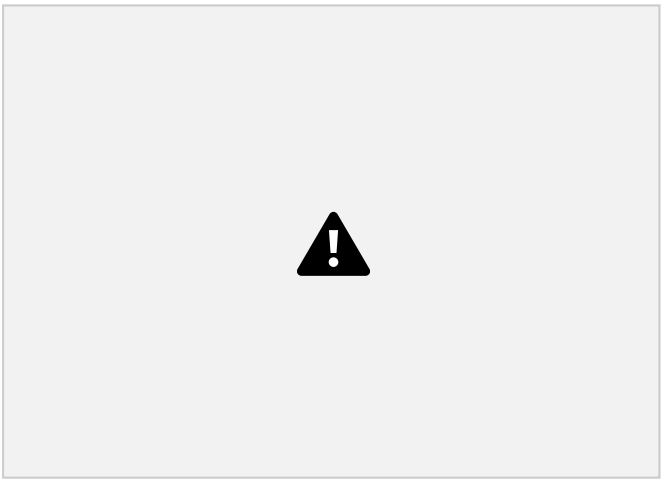


Table 1. Initial distances stored at each node(global view).

Information Stored at Node	Distance to Reach Node A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

We can represent each node's knowledge about the distances to all other nodes as a table like the one given in Table 1.

Note that each node only knows the information in one row of the table.

1. Every node sends a message to its directly connected neighbors containing its personal list of distance. information to its (for example, **A** sends its neighbors **B,C,E**, and **F**.)
2. If any of the recipients of the information from **A** find that **A** is advertising a path shorter than the one they currently know about, they update their list to give the new path length and note that they should send packets for that destination

through **A**. (node **B** learns from **A** that node **E** can be reached at a cost of 1; **B** also knows it can reach **A** at a cost of 1, so it adds these to get the cost of reaching **E** by means of **A**. **B** records that it can reach **E** at a cost of 2 by going through **A**.)

3. After every node has exchanged a few updates with its directly connected neighbors, all nodes will know the least-cost path to all the other nodes.
4. In addition to updating their list of distances when they receive updates, the nodes need to keep track of which node told them about the path that they used to calculate the cost, so that they can create their forwarding table. (for example, **B** knows that it was **A** who said " I can reach **E** in one hop" and so **B** puts an entry in its table that says " To reach **E**, use the link to **A**.)

Table 2. final distances stored at each node (global view).

Information Distance to Reach Node							
Stored at Node	**A**	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	1	0	3	2	1	
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	3	2	1	3	1	0	

In practice, each node's forwarding table consists of a set of triples of the form:

(Destination, Cost, NextHop).

For example, Table 3 shows the complete routing table maintained at node B for the network in figure1.

Table 3. Routing table maintained at node B.

Destination Cost NextHop A 1 A

C 1 C

D 2 C

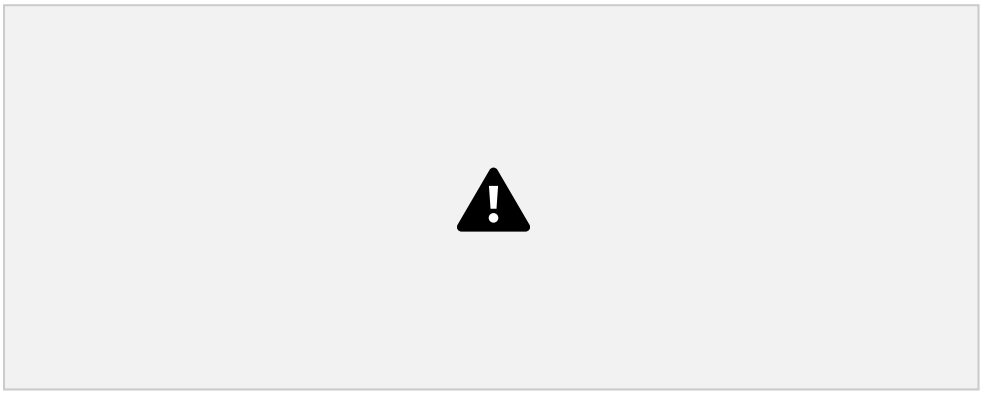
E 2 A

F 2 A

G 3 A

THE COUNT-TO-INFINITY PROBLEM

The count-to-infinity problem.



Consider the five-node (linear) subnet of Fig. 5-10, where the delay metric is the number of hops. Suppose *A* is down initially and all the other routers know this. In other words, they have all recorded the delay to *A* as infinity.

Now let us consider the situation of Fig. 5-10(b), in which all the lines and routers are initially up. Routers *B*, *C*, *D*, and *E* have distances to *A* of 1, 2, 3, and 4, respectively. Suddenly *A* goes down, or alternatively, the line between *A* and *B* is cut, which is effectively the same thing from *B*'s point of view.

LINK STATE ROUTING

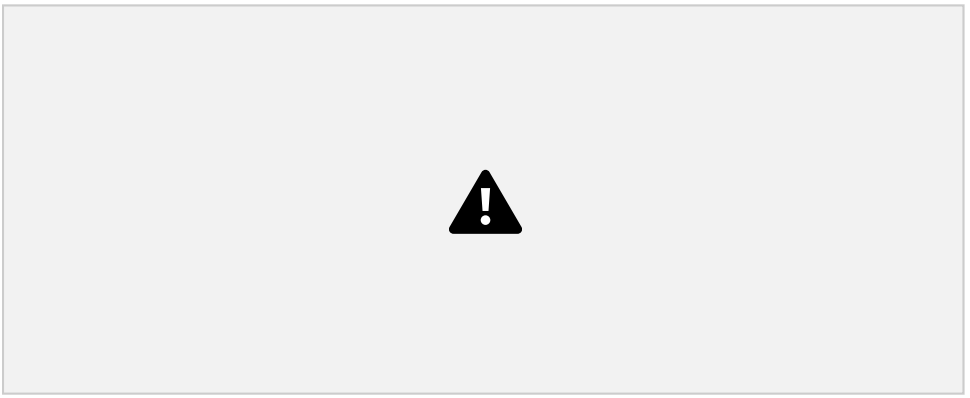
The idea behind link state routing is simple and can be stated as five parts. Each router

must do the following:

1. Discover its neighbors and learn their network addresses.
2. Measure the delay or cost to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to all other routers.
5. Compute the shortest path to every other router

Learning about the Neighbours

When a router is booted, its first task is to learn who its neighbours are. It accomplishes this goal by sending a special HELLO packet on each point-to-point line. The router on the other end is expected to send back a reply telling who it is.



(a) Nine routers and a LAN. (b) A graph model of (a).

(b)

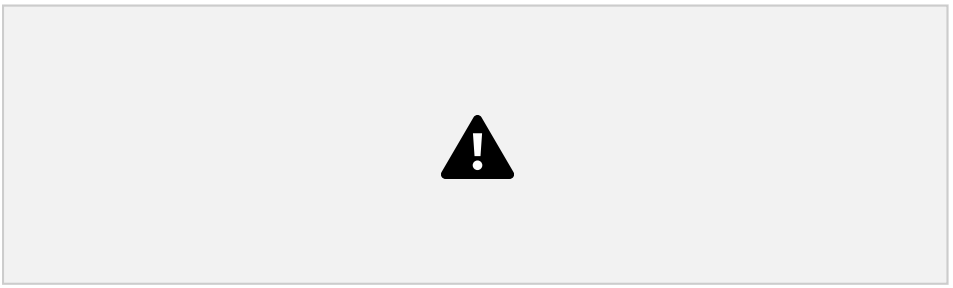
Measuring Line Cost

- The link state routing algorithm requires each router to know, or at least have a reasonable estimate of, the delay to each of its neighbors. The most direct way to determine this delay is to send over the line a special ECHO packet that the other side is required to send back immediately.
- By measuring the round-trip time and dividing it by two, the sending router can get a reasonable estimate of the delay.
- For even better results, the test can be conducted several times, and the average used. Of course, this method implicitly assumes the delays are symmetric, which may not always be the case.

Figure: A subnet in which the East and West parts are connected by two lines.

Unfortunately, there is also an argument against including the load in the delay calculation. Consider the subnet of Fig. 5-12, which is divided into two parts, East and West, connected by two lines, *CF* and *EI*.

Building Link State Packets



(a) A subnet. (b) The link state packets for this subnet.

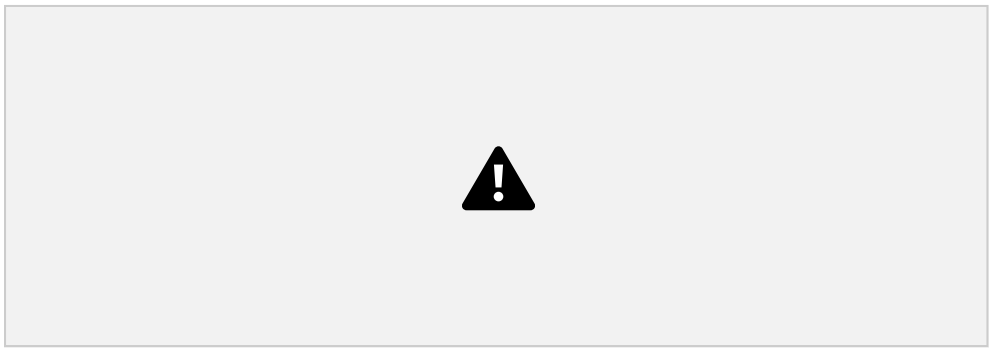
Once the information needed for the exchange has been collected, the next step is for each router to build a packet containing all the data.

The packet starts with the identity of the sender, followed by a sequence number and age (to be described later), and a list of neighbours.

For each neighbour, the delay to that neighbour is given.

An example subnet is given in Fig. 5-13(a) with delays shown as labels on the lines. The corresponding link state packets for all six routers are shown in Fig. 5-13(b).

Distributing the Link State Packets



The packet buffer for router* B *in Fig. 5-13.

In Fig. 5-14, the link state packet from *A* arrives directly, so it must be sent to *C* and *F* and acknowledged to *A*, as indicated by the flag bits.

Similarly, the packet from *F* has to be forwarded to *A* and *C* and acknowledged to *F*.

HIERARCHICAL ROUTING

- The routers are divided into what we will call regions, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions.

- For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations.

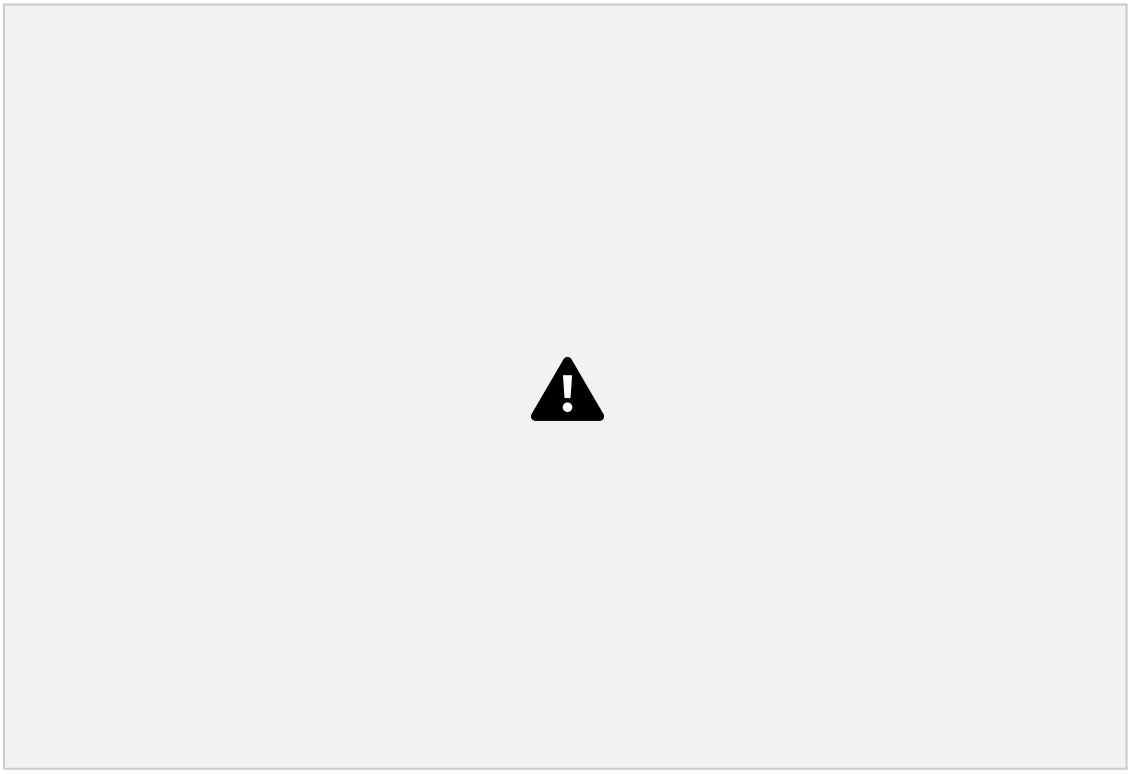


Figure 5-15 gives a quantitative example of routing in a two-level hierarchy with five regions.

The full routing table for router 1*A* has 17 entries, as shown in Fig. 5-15(b). When routing is done hierarchically, as in Fig. 5-15(c), there are entries for all the local routers as before, but all other regions have been condensed into a single router, so all traffic for region 2 goes via the 1*B* -2*A* line, but the rest of the remote traffic goes via the 1*C* -3*B* line.

Hierarchical routing has reduced the table from 17 to 7 entries. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase.

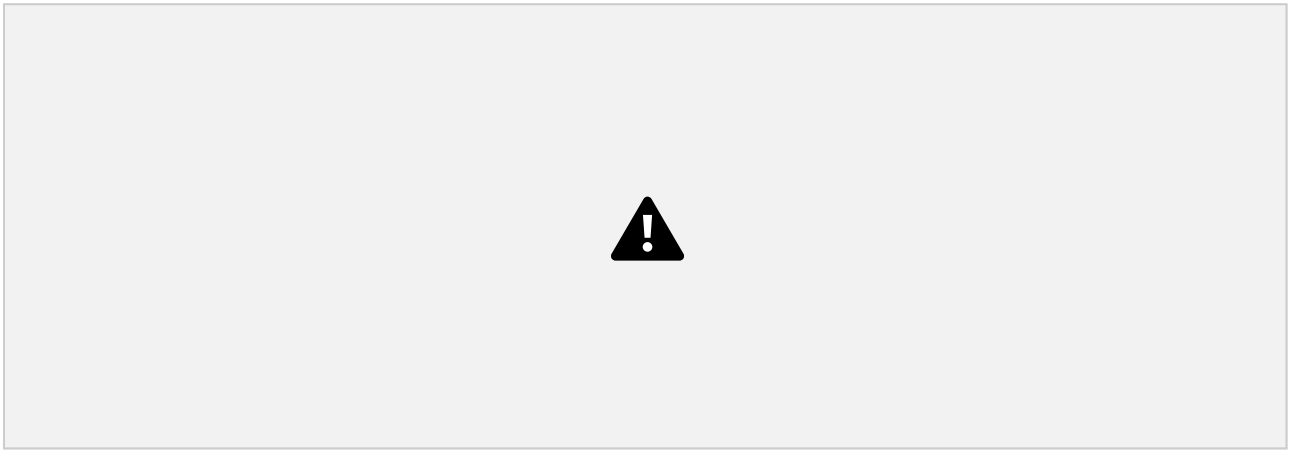
BROADCAST ROUTING

Sending a packet to all destinations simultaneously is called broadcasting.

1) The source simply sends a distinct packet to each destination. Not only is the method wasteful of bandwidth, but it also requires the source to have a complete list of all destinations.

2) Flooding.

The problem with flooding as a broadcast technique is that it generates too many packets and consumes too much bandwidth.



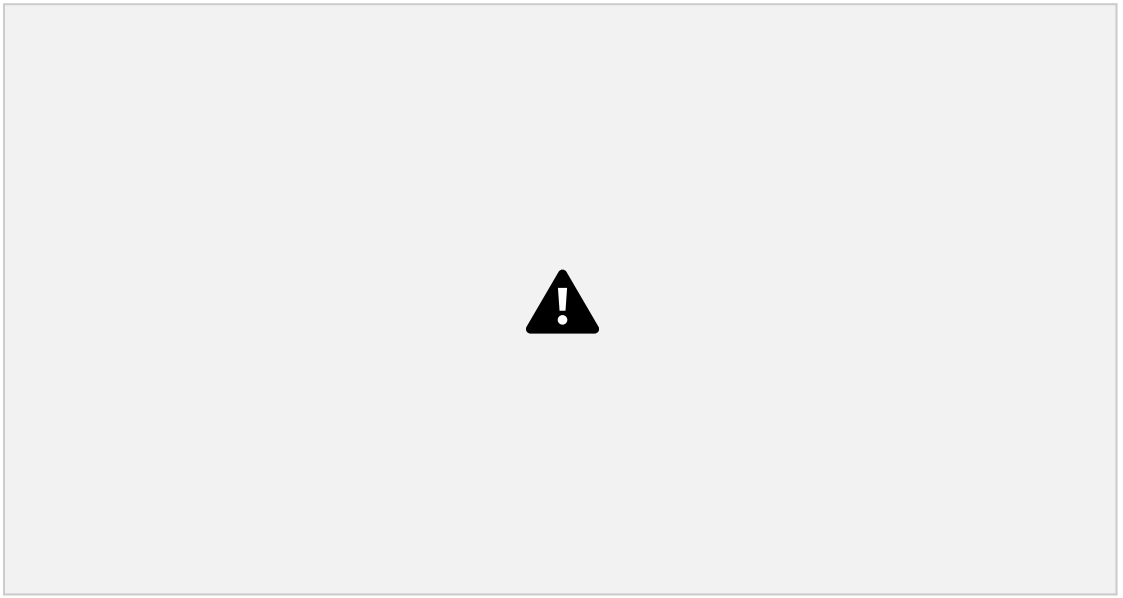
Reverse path forwarding. (a) A subnet. (b) A sink tree. (c) The tree built by reverse path forwarding.

Part (a) shows a subnet, part (b) shows a sink tree for router *I* of that subnet, and part (c) shows how the reverse path algorithm works.

- When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the line that is normally used for sending packets to the source of the broadcast. If so, there is an excellent chance that the broadcast packet itself followed the best route from the router and is therefore the first copy to arrive at the router.

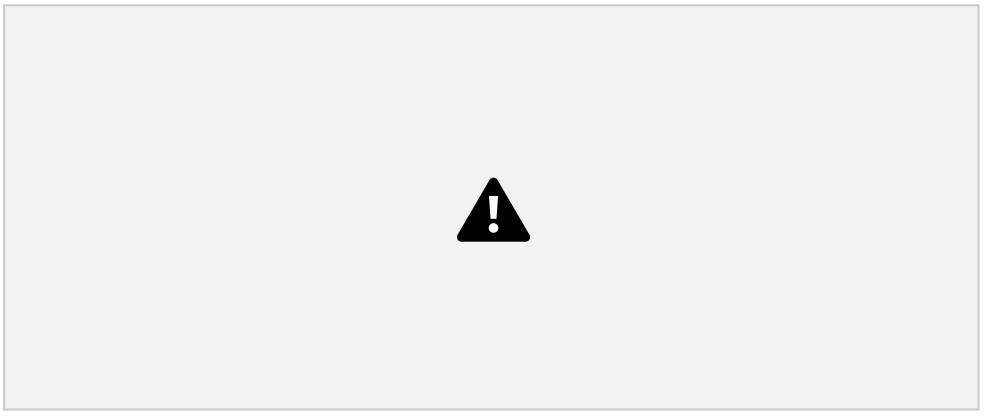
- This being the case, the router forwards copies of it onto all lines except the one it arrived on. If, however, the broadcast packet arrived on a line other than the preferred one for reaching the source, the packet is discarded as a likely duplicate.

MULTICAST ROUTING



- To do multicast routing, each router computes a spanning tree covering all other routers. For example, in Fig. 5-17(a) we have two groups, 1 and 2.
- Some routers are attached to hosts that belong to one or both of these groups, as indicated in the figure.
- A spanning tree for the leftmost router is shown in Fig. 5-17(b). When a process sends a multicast packet to a group, the first router examines its spanning tree and prunes it, removing all lines that do not lead to hosts that are members of the group.
- In our example, Fig. 5-17(c) shows the pruned spanning tree for group 1. Similarly, Fig. 5-17(d) shows the pruned spanning tree for group 2. Multicast packets are forwarded only along the appropriate spanning tree.

ROUTING FOR MOBILE HOSTS



- Hosts that never move are said to be stationary.
- They are connected to the network by copper wires or fiber optics. In contrast, we can distinguish two other kinds of hosts.

- Migratory hosts are basically stationary hosts who move from one fixed site to another from time to time but use the network only when they are physically connected to it.
- Roaming hosts actually compute on the run and want to maintain their connections as they move around.
- We will use the term **mobile hosts** to mean either of the latter two categories, that is, all hosts that are away from home and still want to be connected

The registration procedure typically works like this:

1. Periodically, each foreign agent broadcasts a packet announcing its existence and address. A newly-arrived mobile host may wait for one of these messages, but if none arrives quickly enough, the mobile host can broadcast a packet saying: Are there any foreign agents around?
2. The mobile host registers with the foreign agent, giving its home address, current data link layer address, and some security information.
3. The foreign agent contacts the mobile host's home agent and says: One of your hosts is over here. The message from the foreign agent to the home agent contains the foreign agent's network address. It also includes the security information to convince the home agent that the mobile host is really there.
4. The home agent examines the security information, which contains a timestamp, to prove that it was generated within the past few seconds. If it is happy, it tells the foreign agent to proceed.
5. When the foreign agent gets the acknowledgement from the home agent, it makes an entry in its tables and informs the mobile host that it is now registered.

ROUTING IN AD HOC NETWORKS

We have now seen how to do routing when the hosts are mobile but the routers are fixed. An even more extreme case is one in which the routers themselves are mobile. Among the possibilities are:

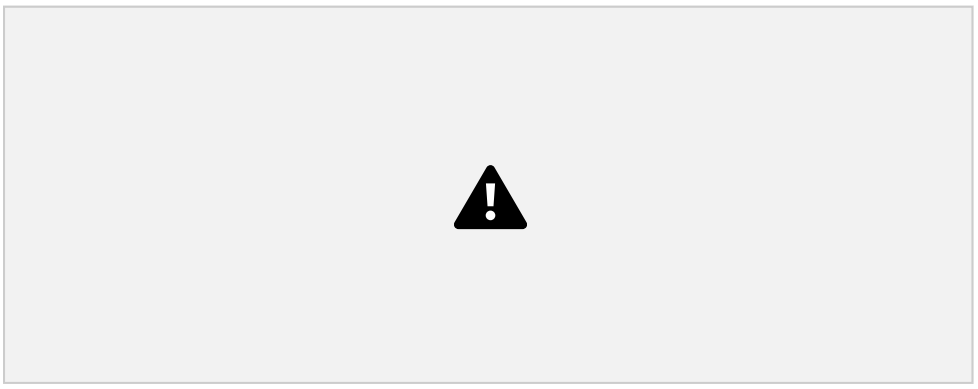
1. Military vehicles on a battlefield with no existing infrastructure.
2. A fleet of ships at sea.
3. Emergency workers at an earthquake that destroyed the infrastructure.
4. A gathering of people with notebook computers in an area lacking 802.11.

In all these cases, and others, each node consists of a router and a host, usually on the same computer. Networks of nodes that just happen to be near each other are called **ad hoc networks** or **MANETs** (**Mobile Ad hoc NETworks**).

- What makes ad hoc networks different from wired networks is that all the usual rules about fixed topologies, fixed and known neighbours, fixed relationship between IP address and location, and more are suddenly tossed out the window.

- Routers can come and go or appear in new places at the drop of a bit. With a wired network, if a router has a valid path to some destination, that path continues to be valid indefinitely (barring a failure somewhere in the system).
- With an ad hoc network, the topology may be changing all the time.
- A variety of routing algorithms for ad hoc networks have been proposed. One of the more interesting ones is the **AODV (Ad hoc On-demand Distance Vector**) routing algorithm (Perkins and Royer, 1999).
- It takes into account the limited bandwidth and low battery life found in environment. Another unusual characteristic is that it is an on-demand algorithm, that is, it determines a route to some destination only when somebody wants to send a packet to that destination. Let us now see what that means.

Route Discovery



(a) Range of* A*'s broadcast. (b) After* B *and* D *have received* A*'s broadcast. (c) After* C*,* F*, and* G *have received* A*'s broadcast. (d) After* E*,* H*, and* I *have received* A*'s broadcast. The shaded nodes are new recipients. The arrows show the possible reverse routes.

- To locate *I*, *A* constructs a special ROUTE REQUEST packet and broadcasts it. The packet reaches *B* and *D*, as illustrated in Fig. 5-20(a).
- The format of the ROUTE REQUEST packet is shown in Fig. 5-21

Format of a ROUTE REQUEST packet.

The format of the ROUTE REQUEST packet is shown in Fig. 5-21. It contains the source and destination addresses, typically their IP addresses, which identify who is looking for whom. It also contains a *Request ID*, which is a local counter maintained separately by each node and incremented each time a ROUTE REQUEST is broadcast. Together, the *Source address* and *Request ID* fields uniquely identify the ROUTE REQUEST packet to allow nodes to discard any duplicates they may receive.

Format of a ROUTE REPLY packet

In addition to the *Request ID* counter, each node also maintains a second sequence counter incremented whenever a ROUTE REQUEST is sent (or a reply to someone else's ROUTE REQUEST). It functions a little bit like a clock and is used to tell new routes from old routes. The fourth field of Fig. 5-21 is *A*'s sequence counter; the fifth field is the most recent value of *I*'s sequence number that *A* has seen (0 if it has never seen it). The use of these fields will become clear shortly. The final field, *Hop count*, will keep track of how many hops the packet has made. It is initialized to 0.

1. No route to *I* is known.

2. The sequence number for *I* in the ROUTE REPLY packet is greater than the value in the routing table.

3. The sequence numbers are equal but the new route is shorter.

CONGESTION CONTROL ALGORITHMS

When too many packets are present in (a part of) the subnet, performance degrades. This situation is called **congestion**.

Figure 5-25 depicts the symptom. When the number of packets dumped into the subnet by the hosts is within its carrying capacity, they are all delivered (except for a few that are afflicted with transmission errors) and the number delivered is proportional to the number sent.

However, as traffic increases too far, the routers are no longer able to cope and they begin losing packets. This tends to make matters worse. At very high traffic, performance collapses completely and almost no packets are delivered.

Figure 5-25. When too much traffic is offered, congestion sets in and performance degrades sharply.

- Congestion can be brought on by several factors. If all of a sudden, streams of packets begin arriving on three or four input lines and all need the same output line, a queue will build up.
- If there is insufficient memory to hold all of them, packets will be lost.
- Slow processors can also cause congestion. If the routers' CPUs are slow at performing the bookkeeping tasks required of them (queuing buffers, updating tables, etc.), queues can build up, even though there is excess line capacity. Similarly, low-bandwidth lines can also cause congestion.

APPROACHES TO CONGESTION CONTROL

- Many problems in complex systems, such as computer networks, can be viewed from a control theory point of view. This approach leads to dividing all solutions into two groups: open loop and closed loop.

Figure: Timescales Of Approaches To Congestion Control

- Open loop solutions attempt to solve the problem by good design.
- Tools for doing open-loop control include deciding when to accept new traffic, deciding when to discard packets and which ones, and making scheduling decisions at various points in the network.
- Closed loop solutions are based on the concept of a feedback loop.

This approach has three parts when applied to congestion control:

1. Monitor the system to detect when and where congestion occurs.
2. Pass this information to places where action can be taken.
3. Adjust system operation to correct the problem.

A variety of metrics can be used to monitor the subnet for congestion. Chief among these are the percentage of all packets discarded for lack of buffer space, the average queue lengths, the number of packets that time out and are retransmitted, the average packet delay, and the standard deviation of packet delay. In all cases, rising numbers indicate growing congestion.

The second step in the feedback loop is to transfer the information about the congestion from the point where it is detected to the point where something can be done about it.

In all feedback schemes, the hope is that knowledge of congestion will cause the hosts to take appropriate action to reduce the congestion.

The presence of congestion means that the load is (temporarily) greater than the resources (in part of the system) can handle. Two solutions come to mind: increase the resources or decrease the load.

CONGESTION PREVENTION POLICIES

The methods to control congestion by looking at open loop systems. These systems are designed to minimize congestion in the first place, rather than letting it happen and reacting after the fact. They try to achieve their goal by using appropriate policies at various levels. In Fig. 5-26 we see different data link, network, and transport policies that can affect congestion (Jain, 1990).

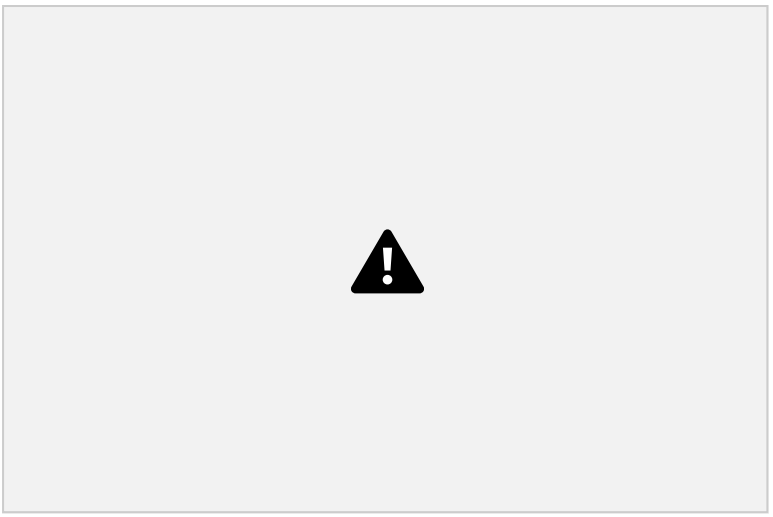


Figure 5-26. Policies that affect congestion.

The **data link layer Policies.**

- The **retransmission policy** is concerned with how fast a sender times out and what it transmits upon timeout. A jumpy sender that times out quickly and retransmits all outstanding packets using go back n will put a heavier load on the system than will a leisurely sender that uses selective repeat.

- Closely related to this is the **buffering policy**. If receivers routinely discard all out-of order packets, these packets will have to be transmitted again later, creating extra load. With respect to congestion control, selective repeat is clearly better than go back n.

- **Acknowledgement policy** also affects congestion. If each packet is acknowledged if immediately, the acknowledgement packets generate extra traffic. However, acknowledgements are saved up to piggyback onto reverse traffic, extra timeouts and retransmissions may result. A tight flow control scheme (e.g., a small window) reduces the data rate and thus helps fight congestion.

The **network layer Policies.**

- The choice between using **virtual circuits and using datagrams** affects congestion since many congestion control algorithms work only with virtual-circuit subnets.

- **Packet queueing and service policy** relates to whether routers have one queue per input line, one queue per output line, or both. It also relates to the order in which packets are processed (e.g., round robin or priority based).

- **Discard policy** is the rule telling which packet is dropped when there is no space.

- A good **routing algorithm** can help avoid congestion by spreading the traffic over all the lines, whereas a bad one can send too much traffic over already congested lines.

- **Packet lifetime management** deals with how long a packet may live before being discarded. If it is too long, lost packets may clog up the works for a long time, but if it is too short, packets may sometimes time out before reaching their destination, thus inducing retransmissions.

The **transport layer Policies**,

- The **same issues occur as in the data link layer**, but in addition, determining the **timeout interval** is harder because the transit time across the network is less predictable than the transit time over a wire between two routers. If the timeout interval is too short, extra packets will be sent unnecessarily. If it is too long, congestion will be reduced but the response time will suffer whenever a packet is lost.

ADMISSION CONTROL

- One technique that is widely used to keep congestion that has already started from getting worse is **admission control**.

- Once congestion has been signaled, no more virtual circuits are set up until the problem has gone away.

- An alternative approach is to allow new virtual circuits but carefully route all new virtual circuits around problem areas. For example, consider the subnet of Fig. 5-27(a), in which two routers are congested, as indicated.

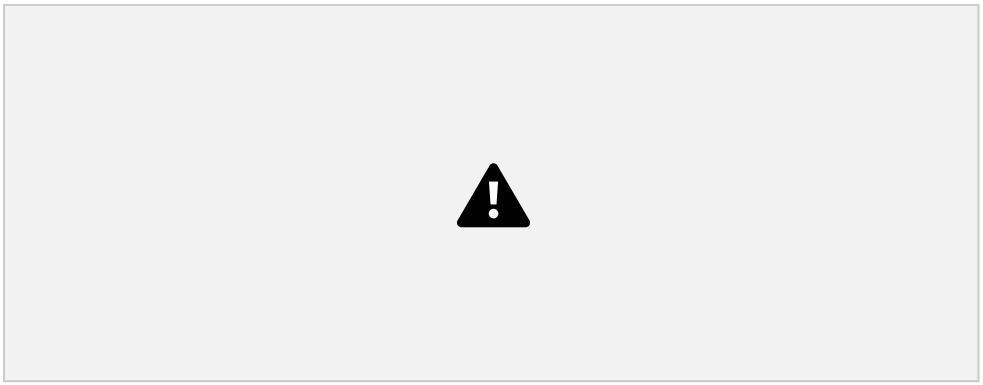


Figure 5-27. (a) A congested subnet. (b) A redrawn subnet that eliminates the congestion. A virtual circuit from* A *to* B *is also shown.

Suppose that a host attached to router *A* wants to set up a connection to a host attached to router *B*. Normally, this connection would pass through one of the congested routers. To avoid this situation, we can redraw the subnet as shown in Fig. 5-27(b), omitting the congested routers and all of their lines. The dashed line shows a possible route for the virtual circuit that avoids the congested routers.

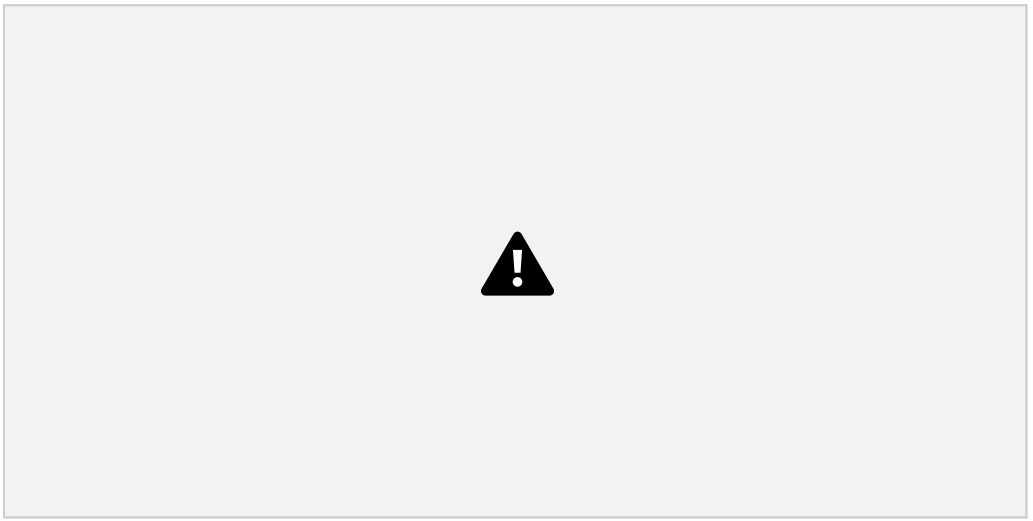
TRAFFIC AWARE ROUTING

These schemes adapted to changes in topology, but not to changes in load. The goal in taking load into account when computing routes is to shift traffic away from hotspots

that will be the first places in the network to experience congestion.

The most direct way to do this is to set the link weight to be a function of the (fixed) link bandwidth and propagation delay plus the (variable) measured load or average queuing delay. Least-weight paths will then favor paths that are more lightly loaded, all else being equal.

Consider the network of Fig. 5-23, which is divided into two parts, East and West, connected by two links, *CF* and *EI*. Suppose that most of the traffic between East and West is using link *CF*, and, as a result, this link is heavily loaded with long delays. Including queueing delay in the weight used for the shortest path calculation will make *EI* more attractive. After the new routing tables have been installed, most of the East-West traffic will now go over *EI*, loading this link. Consequently, in the next update, *CF* will appear to be the shortest path. As a result, the routing tables may oscillate wildly, leading to erratic routing and many potential problems.



If load is ignored and only bandwidth and propagation delay are considered, this problem does not occur. Attempts to include load but change weights within a narrow range only slow down routing oscillations. Two techniques can contribute to a successful solution. The first is multipath routing, in which there can be multiple paths from a source to a destination. In our example this means that the traffic can be spread across both of the East to West links. The
second one is for the routing scheme to shift traffic across routes slowly enough that it is able to converge.

TRAFFIC THROTTLING

Each router can easily monitor the utilization of its output lines and other resources. For example, it can associate with each line a real variable, *u*, whose value, between 0.0 and

1.0, reflects the recent utilization of that line. To maintain a good estimate of *u*, a

sample of the instantaneous line utilization, f (either 0 or 1), can be made periodically and u updated according to

where the constant a determines how fast the router forgets recent history.

Whenever u moves above the threshold, the output line enters a "warning" state. Each newly arriving packet is checked to see if its output line is in warning state. If it is, some action is taken. The action taken can be one of several alternatives, which we will now discuss.

THE WARNING BIT

The old DECNET architecture signaled the warning state by setting a special bit in the packet's header.

When the packet arrived at its destination, the transport entity copied the bit into the next acknowledgement sent back to the source. The source then cut back on traffic.

As long as the router was in the warning state, it continued to set the warning bit, which meant that the source continued to get acknowledgements with it set.

The source monitored the fraction of acknowledgements with the bit set and adjusted its transmission rate accordingly. As long as the warning bits continued to flow in, the source continued to decrease its transmission rate. When they slowed to a trickle, it increased its transmission rate.

Note that since every router along the path could set the warning bit, traffic increased only when no router was in trouble.

CHOKE PACKETS

In this approach, the router sends a **choke packet** back to the source host, giving it the destination found in the packet.

The original packet is tagged (a header bit is turned on) so that it will not generate any more choke packets farther along the path and is then forwarded in the usual way.

When the source host gets the choke packet, it is required to reduce the traffic sent to the specified destination by X percent. Since other packets aimed at the same destination are probably already under way and will generate yet more choke packets, the host should ignore choke packets referring to that destination for a fixed time interval. After that period has expired, the host listens for more choke packets for another interval. If one

arrives, the line is still congested, so the host reduces the flow still more and begins ignoring choke packets again. If no choke packets arrive during the

listening period, the host may increase the flow again.

- The feedback implicit in this protocol can help prevent congestion yet not throttle any flow unless trouble occurs.
- Hosts can reduce traffic by adjusting their policy parameters.
- Increases are done in smaller increments to prevent congestion from reoccurring quickly.
- Routers can maintain several thresholds. Depending on which threshold has been crossed, the choke packet can contain a mild warning, a stern warning, or an ultimatum.

HOP-BY-HOP BACK PRESSURE

- At high speeds or over long distances, sending a choke packet to the source hosts does not work well because the reaction is so slow.

Consider, for example, a host in San Francisco (router *A* in Fig. 5-28) that is sending traffic to a host in New York (router *D* in Fig. 5-28) at 155 Mbps. If the New York host begins to run out of buffers, it will take about 30 msec for a choke packet to get back to San Francisco to tell it to slow down. The choke packet propagation is shown as the second, third, and fourth steps in Fig. 5-28(a). In those 30 msec, another 4.6 megabits will have been sent. Even if the host in San Francisco completely shuts down immediately, the 4.6 megabits in the pipe will continue to pour in and have to be dealt with. Only in the seventh diagram in Fig. 5- 28(a) will the New York router notice a slower flow.

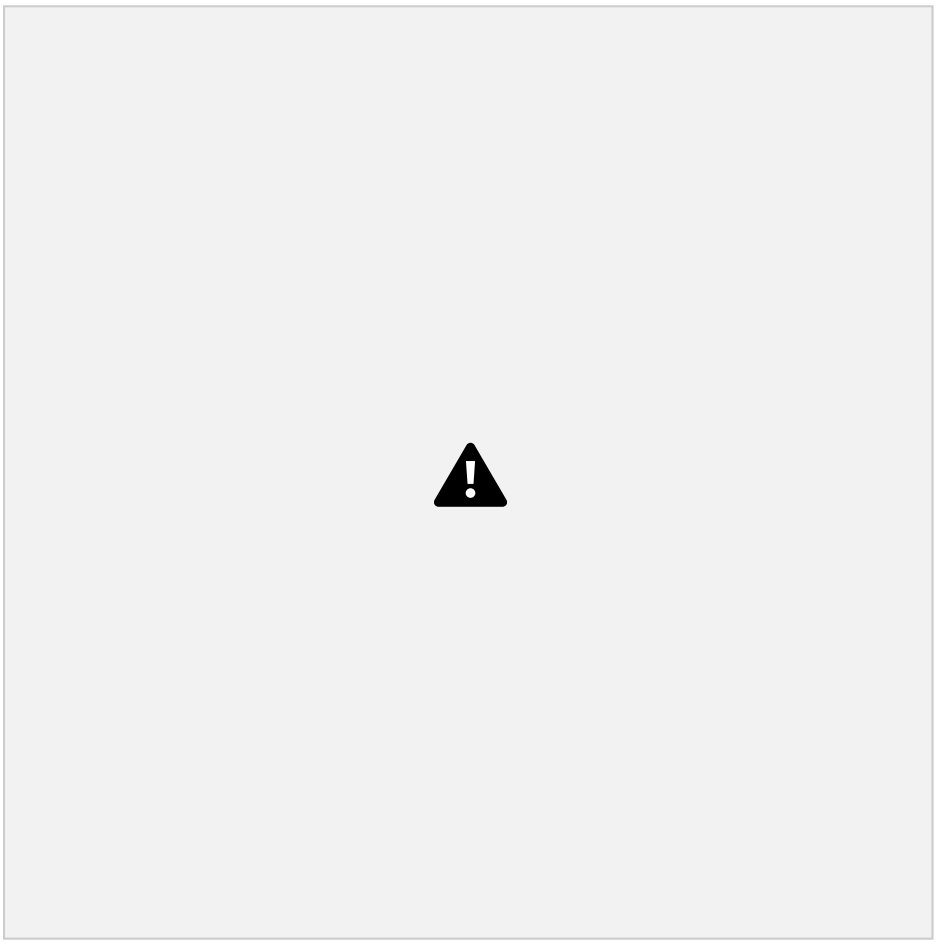


Figure 5-28. (a) A choke packet that affects only the source. (b) A choke packet that affects each hop it passes through.

An alternative approach is to have the choke packet take effect at every hop it passes through, as shown in the sequence of Fig. 5-28(b). Here, as soon as the choke packet reaches *F*, *F* is required to reduce the flow to *D*. Doing so will require *F* to devote more buffers to the flow, since the source is still sending away at full blast, but it gives *D* immediate relief, like a headache remedy in a television commercial. In the next step, the choke packet reaches *E*, which tells *E* to reduce the flow to *F*. This action puts a greater demand on *E*'s buffers but gives *F* immediate relief. Finally, the choke packet reaches *A* and the flow genuinely slows down.

The net effect of this hop-by-hop scheme is to provide quick relief at the point of congestion at the price of using up more buffers upstream. In this way, congestion can be nipped in the bud without losing any packets.

LOAD SHEDDING

- When none of the above methods make the congestion disappear, routers can bring out the heavy artillery: load shedding.

- **Load shedding** is a fancy way of saying that when routers are being in undated by packets that they cannot handle, they just throw them away.

- A router drowning in packets can just pick packets at random to drop, but usually it can do better than that.
- Which packet to discard may depend on the applications running.
- To implement an intelligent discard policy, applications must mark their packets in priority classes to indicate how important they are. If they do this, then when packets have to be discarded, routers can first drop packets from the lowest class, then the next lowest class, and so on.

RANDOM EARLY DETECTION

- It is well known that dealing with congestion after it is first detected is more effective than letting it gum up the works and then trying to deal with it. This observation leads to

  the idea of discarding packets before all the buffer space is really exhausted. A popular algorithm for doing this is called **RED** (**Random Early Detection**).
- In some transport protocols (including TCP), the response to lost packets is for the source to slow down. The reasoning behind this logic is that TCP was designed for wired networks and wired networks are very reliable, so lost packets are mostly due to buffer overruns rather than transmission errors. This fact can be exploited to help **reduce congestion**.
- By having routers drop packets before the situation has become hopeless (hence the "early" in the name), the idea is that there is time for action to be taken before it is too late. To determine when to start discarding, routers maintain a running average of their queue lengths. When the average queue length on some line exceeds a threshold, the line is said to be congested and action is taken.

JITTER CONTROL

- The variation (i.e., standard deviation) in the packet arrival times is called **jitter**.
- High jitter, for example, having some packets taking 20 msec and others taking 30 msec to arrive will give an uneven quality to the sound or movie. Jitter is illustrated in Fig. 5- 29. In contrast, an agreement that 99 percent of the packets be delivered with a delay in the range of 24.5 msec to 25.5 msec might be acceptable.

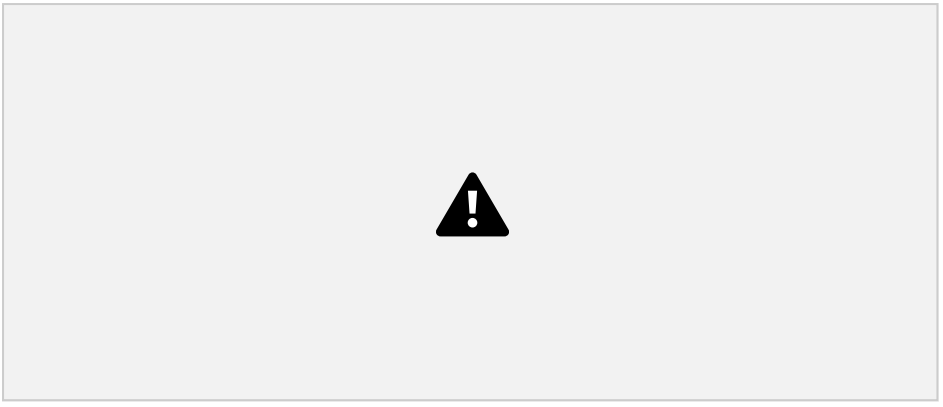


Figure 5-29. (a) High jitter. (b) Low jitter.

The jitter can be bounded by computing the expected transit time for each hop along the path. When a packet arrives at a router, the router checks to see how much the packet is

behind or ahead of its schedule. This information is stored in the packet and updated at each hop. If the packet is ahead of schedule, it is held just long enough to get it back on schedule. If it is behind schedule, the router tries to get it out the door quickly.

In fact, the algorithm for determining which of several packets competing for an output line should go next can always choose the packet furthest behind in its schedule.

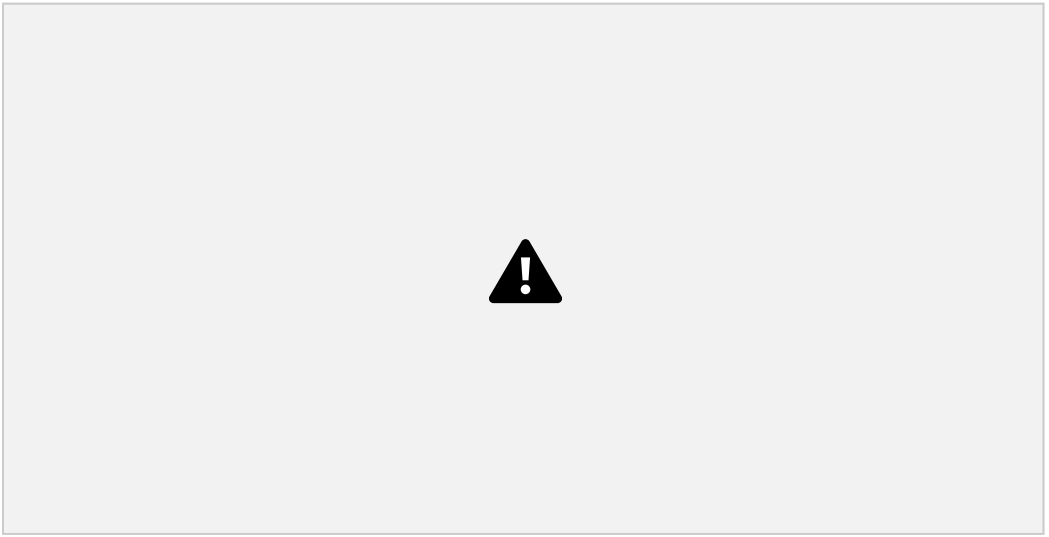
In this way, packets that are ahead of schedule get slowed down and packets that are behind schedule get speeded up, in both cases reducing the amount of jitter.

In some applications, such as video on demand, jitter can be eliminated by buffering at the receiver and then fetching data for display from the buffer instead of from the

network in real time. However, for other applications, especially those that require real time interaction between people such as Internet telephony and videoconferencing, the delay inherent in buffering is not acceptable.

How to correct the Congestion Problem:

Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. Congestion control mechanisms are divided into two categories, one category prevents the congestion from happening and the other category removes congestion after it has taken place.



These two categories are:

1. Open loop

2. Closed loop

Open Loop Congestion Control

• In this method, policies are used to prevent the congestion before it happens.

• Congestion control is handled either by the source or by the destination.

• The various methods used for open loop congestion control are:

1. Retransmission Policy

• The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.

• However retransmission in general may increase the congestion in the network. But we need to implement good retransmission policy to prevent congestion.

• The retransmission policy and the retransmission timers need to be designed to optimize efficiency and at the same time prevent the congestion.

2. Window Policy

• To implement window policy, selective reject window method is used for congestion control.

• Selective Reject method is preferred over Go-back-n window as in Go-back-n method, when timer for a packet times out, several packets are resent, although some may have arrived safely at the receiver. Thus, this duplication may make congestion worse.

• Selective reject method sends only the specific lost or damaged packets.

3. Acknowledgement Policy

• The acknowledgement policy imposed by the receiver may also affect congestion.

• If the receiver does not acknowledge every packet it receives it may slow down the sender and help prevent congestion.

• Acknowledgments also add to the traffic load on the network. Thus, by sending fewer acknowledgements we can reduce load on the network.

• To implement it, several approaches can be used:

1. A receiver may send an acknowledgement only if it has a packet to be sent.

2. A receiver may send an acknowledgement when a timer expires.

3. A receiver may also decide to acknowledge only *N* packets at a time.

4. Discarding Policy

• A router may discard less sensitive packets when congestion is likely to happen.

• Such a discarding policy may prevent congestion and at the same time may not harm the integrity of the transmission.

5. Admission Policy

• An admission policy, which is a quality-of-service mechanism, can also prevent congestion in virtual circuit networks.

• Switches in a flow first check the resource requirement of a flow before admitting it to the network.

• A router can deny establishing a virtual circuit connection if there is congestion in the "network or if there is a possibility of future congestion.

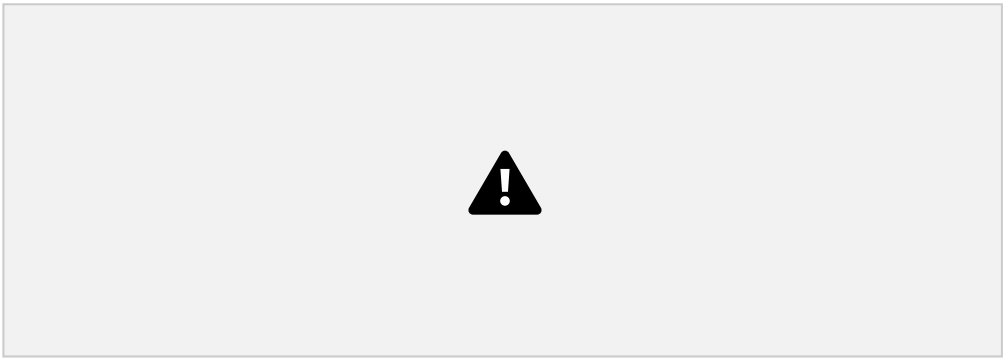
Closed Loop Congestion Control

• Closed loop congestion control mechanisms try to remove the congestion after it

happens. • The various methods used for closed loop congestion control are:

1. Backpressure

• Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow.



• The backpressure technique can be applied only to virtual circuit networks. In such virtual circuit each node knows the upstream node from which a data flow is coming.

• In this method of congestion control, the congested node stops receiving data from the immediate upstream node or nodes.

• This may cause the upstream node on nodes to become congested, and they, in turn, reject data from their upstream node or nodes.

• As shown in fig node 3 is congested and it stops receiving packets and informs its upstream node 2 to slow down. Node 2 in turns may be congested and informs node 1 to slow down. Now node 1 may create congestion and informs the source node to slow down. In this way
the congestion is alleviated. Thus, the pressure on node 3 is moved backward to the source to remove the congestion.

2. Choke Packet

• In this method of congestion control, congested router or node sends a special type of packet called choke packet to the source to inform it about the congestion.

• Here, congested node does not inform its upstream node about the congestion as in backpressure method.

• In choke packet method, congested node sends a warning directly to the source station *i.e.* the intermediate nodes through which the packet has traveled are not warned.

3. Implicit Signaling

• In implicit signaling, there is no communication between the congested node or nodes and the source.

• The source guesses that there is congestion somewhere in the network when it does not receive any acknowledgment. Therefore the delay in receiving an acknowledgment is interpreted as congestion in the network.

• On sensing this congestion, the source slows down.

• This type of congestion control policy is used by TCP.

4. Explicit Signaling

• In this method, the congested nodes explicitly send a signal to the source or

destination to inform about the congestion.

• Explicit signaling is different from the choke packet method. In choke packed method, a separate packet is used for this purpose whereas in explicit signaling method, the signal is included in the packets that carry data .

• Explicit signaling can occur in either the forward direction or the backward direction .

• In backward signaling, a bit is set in a packet moving in the direction opposite to the congestion. This bit warns the source about the congestion and informs the source to slow down.

• In forward signaling, a bit is set in a packet moving in the direction of congestion. This bit warns the destination about the congestion. The receiver in this case uses policies such as slowing down the acknowledgements to remove the congestion.