Date 9 7 24
Page No.

Computer Network

* Advantages of Computer Network :-

① Resource Sharing :- Resource sharing is one of the important applications of computer network. You can share a single software among multiple users. We can also share Hardware Devices via this technique.

② Business Applications :- The goal is to make all programs, equipment, and especially data available to anyone on the network without regard to the physical location of the resource or the user. For example; :- Having a group of office workers share a common printer.

③ Scalability :- Businesses can quickly expand their infrastructure using computer networks to accommodate more users, divices and data without experiencing significant disruptions.

④ Increases Cost Efficiency :→ Networking technology is constantly improving, which has prompted the creation of affordable Software and apps. Organizations can centralize their IT infrastructure instead of acquiring separate resources for each user.

(5) Reliability :- Reliability implies backing up informations.

(5) Real-time Updates and information Dissemination :-
Computer networks provide real-time updates and information dissemination in education, healthcare, and news media industries.

* Disadvantages of Computer Network :-

(1) Security Concerns :- One of the most serious drawbacks of computer networks is the increased danger of security breaches. Because networks are interconnected, vulnerabilities exist, hackers can get unauthorized access to sensitive data, disrupt services.

(2) Complexity and Upkeep :- Managing and maintaining computer networks may take time and effort.

(3) Bandwidth Limitations :- Despite developments in network architecture, bandwidth constraints can still be a problem, particularly in locations with inadequate internet connectivity. Video conferencing, huge file transfers and online gaming can quickly use available bandwidth, reducing overall network speed.

(4) Malware Propagation Possibility :→ Computer networks allow malware to spread quickly. A single infected device can infect a whole networks, resulting in compromised systems.

* Computer Network :→ A computer network is a web of interconnected devices communicating and exchanging resources in real time.

Two computers are said to be interconnected if they are able to exchange information. The connection need not be via a copper wire; fiber optics, microwaves, infrared, and communication satelites can also be used. Networks come in many sizes, shapes and forms. They are usually connected together to make large networks.

* Applications of Computer Network :-

(i) Resource sharing (ii) Communication. (iii) Business Application
(iv) Cloud Computing (v) Virtualization (vi) Remote Working
(vii) Real-time collaboration.

* Network Hardware :→ There are two types of transmission technology that are in widespread use : (i) Broadcast Link, (ii) Point-to-Point links.

(i) Point-to-Point links :→ Point-to-point links connect individual pairs of machines. To go from the source to the destination on a network made up of point-to-point links, short messages, called packets in certain contexts, may have to first visit one or more intermediate machines. Often multiple routes, of different lengths, are possible, so finding good ones is important in point-to-point networks.

(ii) Broadcast Network :→ On a broadcast network, the communication channel is shared by all the machines on the network; packets sent by any machine are received by all the others. An address field within each packet specifies the intended recipient. Upon receiving a packet, a machine checks the address field. If the packet is intended for the receiving machine, that machine processes the packet.

A wireless network is a common example of a broadcast link. Broadcast systems usually also allow the possibility of addressing a packet to all destinations by using a special code in the address fields.

When a packet with this code is transmitted, it is received and processed by every machine on the network. This mode of operation is called broadcasting.

Imp

* How a distributed system is different from computer networks :-

The key distinction is that in a distributed system, a collection of independent computers appears to its users as a single coherent system. Usually, it has a single model or paradigm that it presents to the users. Often a layer of software on top of the operating system, called middleware, is responsible for implementing this model.

In a computer network, this coherence, model, and software are absent. Users are exposed to the actual machines, without any attempt by the system to make the machines look and act in a coherent way. If the machines have different hardware and different operating systems, that is fully visible to the users.

In effect, a distributed system is a software system built on top of a network. Thus, the distinction between a network and a distributed system lies with the software, rather than with the

hardware.

⇒ An alternative criterion for classifying networks is by scale.

(10 m)

(1.) Personal Area Networks :- PANs let devices communicate over the range of a person. A common example is a wireless network that connects a computer with its peripherals. Without using wireless, this connection must be done with cables. To help these users, some companies got design a short-range wireless network called Bluetooth to connect these components without wires.

(10 m – 1 km)

(2.) Local Area Networks :- A LAN is a privately owned network that operates within and nearby a single building like a home, office or factory. LANs are widely used to connect personal computers and consumer electronics to let them share resources (e.g., Printers). When LANs are used by companies, they are called enterprise networks.

There is a standard for wireless LANs called IEEE 802.11, popularly known as WiFi, which has become very widespread. The topology of many wired LANs is built from point-to-point links.

Date
Page No.

③ Metropolitan Area Networks :- A MAN covers a city. The best-known example of MANs are the cable television networks available in many cities. These systems grew from earlier community antenna systems used in areas with poor over-the-air television reception.

At first, these were locally designed, adhoc systems. Then companies began jumping into the business, getting contracts from local governments to wire up entire cities.

④ Wide Area Networks :- A WAN spans a large geographical area, often a country or continent.

The WAN is a network that connects offices in ~~three~~ different cities. Each of these offices contains computers intended for running user programs. We call these machines hosts. The rest of the network that connects these hosts is then called subnet. The job of the subnet is to carry messages from host to host.

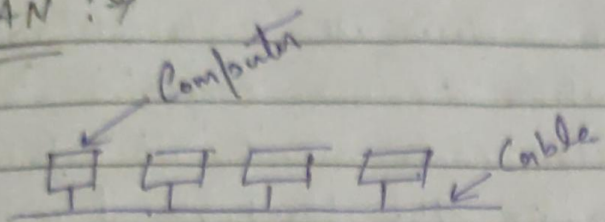
In most WANs, the subnet consists of two distinct components: transmission lines and switching elements.

Transmission lines move bits between machines. They can be made of copper wire, optical fiber, or even radio links. Most of companies do not have transmission lines lying about, so instead they lease the lines from a telecommunications company.

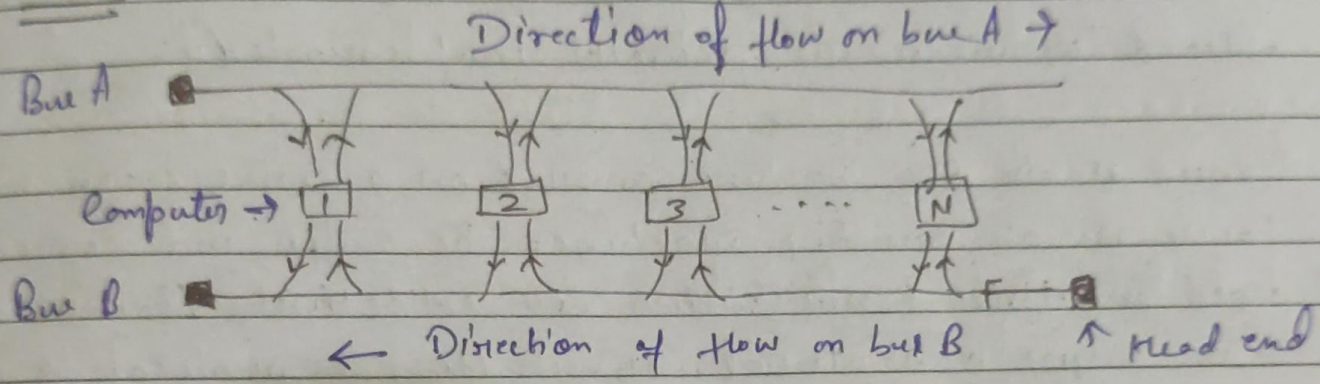
Switching elements, or just switches, are specialized computers that connect two or more transmission lines. When data arrive on an incoming line, the switching element must choose an outgoing line on which to forward them. These machines are also called "routers".

(5) Internetworks : Many networks exist in the world, often with different hardware and software. People connected to one network often want to communicate with people attached to a different one. The fulfillment of this desire requires that different, and frequently incompatible, networks be connected. A collection of interconnected networks is called an internetwork or Internet.

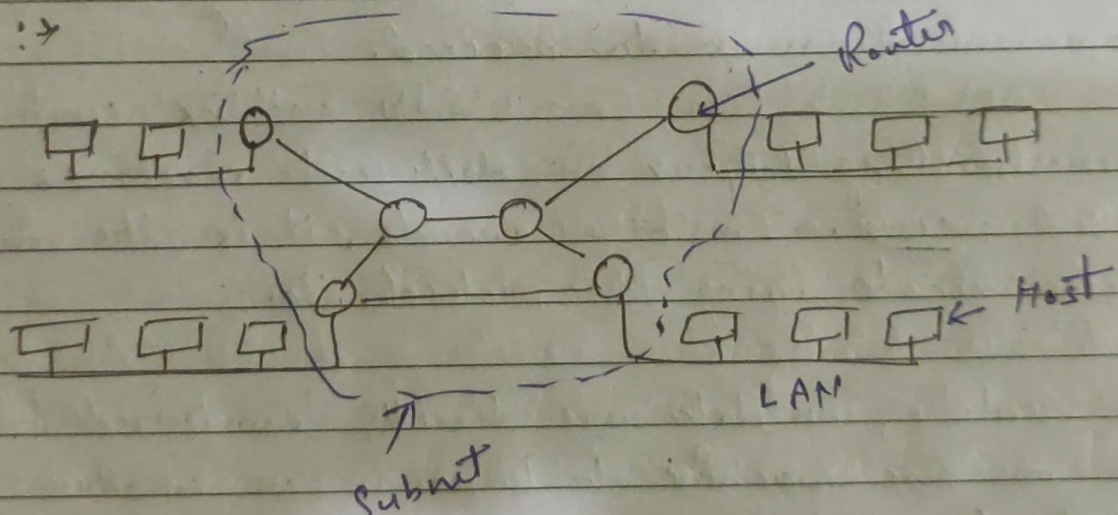
(i) LAN :-



(ii) MAN :-



(iii) WAN :-



Imp

(*) Protocol Hierarchies :- To reduce their design complexity, most networks are organized as a series of layers or levels, each one built upon the one below it. In all networks, the purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented.

Layer n on one machine carries on a conversation with layer n on another machine. The rules and conventions used in this conversation are collectively known as the layer n protocol. Basically, a protocol is an agreement between the communicating parties on how communication is to proceed.

The entities comprising the corresponding layers on different machines are called peers. In other words, it is the peers that communicate using the protocol.

In reality, no data are directly transferred from layer n on one machine to layer n on another machine. Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached. through which actual communication occurs.

* Interface :- Between each pair of adjacent layer there is an interface. The interface defines which primitive operations and services the lower layer offers to the upper layer one.

When network designers decide how many layers to include in a network and what each one should do, one of the most important considerations is defining clear interfaces between the layers.

* Network architecture :- A set of layers and protocols is called a network architecture.

The specification of an architecture must contain enough information to allow an implementer to write the program or build the hardware for each layer so that it will correctly obey the appropriate protocol. Neither the details of the implementation nor the specification of the interface are part of the architecture because these are hidden away inside the machines and not visible from the outside.

* Subtasks in Computer Network :-

① Packetized data ② Common format
③ Binary data to Signals
④ who can access channel when.

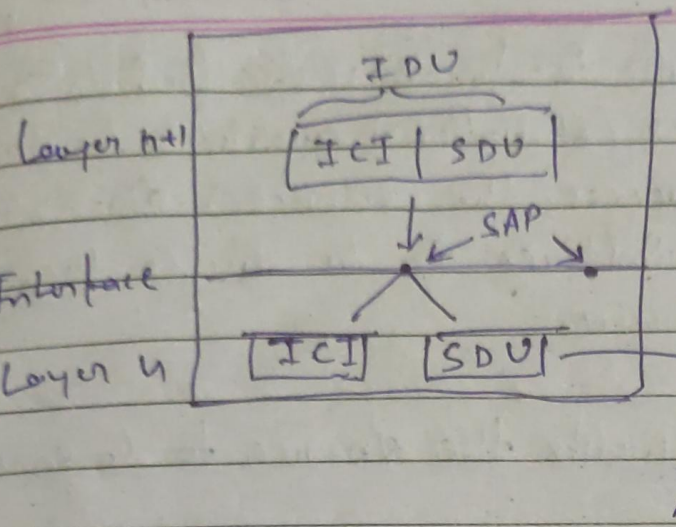
[Service Access Point]

Date
Page No.

* Services in Computer Network :→ Services are available at SAPs (Service Access Points). The layer n SAPs are the places where layer (n+1) can access the services offered. Each SAP has an address that uniquely identifies it. For example, the SAPs in the telephone system are the sockets into which modular telephones can be plugged, and the SAP addresses are the telephone numbers of these sockets.

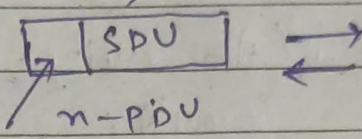
* Interface Data Unit :→ In order for two layers to exchange information, there has to be an agreed upon set of rules about the interface. At a typical interface, the layer (n+1) entity passes an IDU (Interface Data Unit) to the layer n entity through the SAP. The IDU consists of an SDU (Service Data Unit) and some control information. The SDU is the information passed across the network to the peer entity and then up to layer n+1. The control information is needed to help the lower layer do its job.

* Protocol Data Unit :→ In order to transfer the SDU, the layer n entity may have to fragment it into several pieces, each of which is given a header and sent as a separate PDU (Protocol Data Unit) such as packet.



IDU = Interface Data Unit

ICI = Interface Control information.



Layer n entities exchanges n-PDUs in their layer n protocol

Relation between layers at an interface

* Connection-Oriented Services : Connection-oriented service is modeled after the telephone system. To use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releases the connection.

* Connectionless Service : Connectionless service is modeled after the postal system. Each message carries the full destination address, and each one is routed through the system independent of all the others.

* Service Primitives :- A service is formally specified by a set of primitives (operations) available to a user or other entity to access the service. One way to classify the service primitives is to divide them into four classes: as shown

(i) Request :- An entity wants the service to do some work.

(ii) Indication :- An entity is to be informed about an event.

(iii) Response :- An entity wants to respond to an event.

(iv) Confirm :- The response to an earlier request has come back.

* Relationship of Services to protocols :- A service is a set of primitives (operations) that a layer provides to the layer above it. The service defines what operations the layer is prepared to perform on behalf of its users, but it says nothing at all about how these operations are implemented. A service relates to an interface between two layers, with the lower layer being the service provider

and the upper layer being the service user.

A protocol, in contrast, is a set of rules governing the format and meaning of the frames, packets, or messages that are exchanged by the peer entities within a layer. Entities use protocols in order to implement their service definitions. They are free to change their protocols at will, provided they do not change the service visible to their users. In this way, the service and the protocol are completely decoupled.

* The OSI Reference Model: This model is based on a proposal developed by ISO [International Standards Organization] in 1983. The model is called 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 principles that were applied to arrive at the seven layers are as follows:-

1. A layer should be created where a different level of 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.

7 [Application]
6 [Presentation]
5 [Session]
4 [Transport]
3 [Network]
2 [Data link]
1 [Physical]

(1) Physical Layer :- This 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.

(ii) * Data link layer :- The main task of the data link layer is to take a raw transmission facility and transform it into a line that appears free of undetected transmission errors to the network layer.

It accomplishes this task by having the sender break the input data up into data frames, transmit the frames sequentially, and process the acknowledgement frames sent back by the receiver.

(iii) * Network layer :- This layer is concerned with controlling 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.

If too many packets are present in the subnet at the same time, they will get in each other's way, forming bottlenecks. The control of such congestion also belongs to the network layer.

When a packet crosses a national border, with different rates on each side, the accounting can become complicated. The addressing used by the second network may be different from the first one.

(iv)

* Transport Layer :- The basic function of the transport layer is to accept data from the session layer, 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.

The transport layer also determines what type of service to provide the session layer, and ultimately, 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.

Transport layer is a true end-to-end layer, from source to destination.

(v) The Session Layer :- The session layer allows users on different machines to establish sessions between them. A session allows ordinary data transport, as does the transport layer, but it also provides enhanced services useful in some applications.

A related session service is token management. For some protocols, it is essential that both sides do not attempt the same operation at the same time. To manage these activities, the session layer provides tokens that can be exchanged.

Another Session Service is Synchronization.

(vi) Presentation layer :- The presentation layer performs certain functions that are requested sufficiently often to warrant finding a general solution for them, rather than letting each user solve the problems.

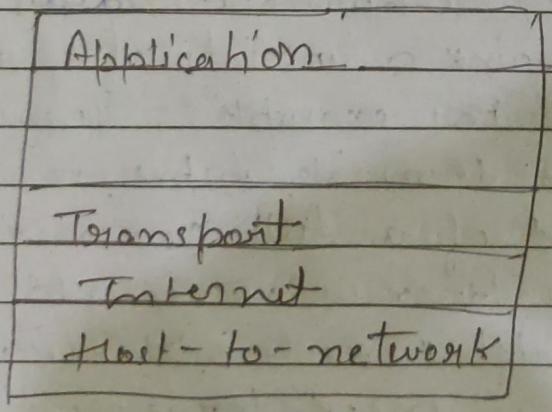
In particular, unlike all the lower layers, which are just interested in moving bits reliably from here to there, the presentation layer is concerned with the syntax and semantics of the information transmitted.

(vii) The Application layer :- The application layer contains a variety of protocols that are commonly needed. For example :- there are hundreds of incompatible terminal types in the world. Consider the plight of a full screen editor that is supposed to work over a network with many different terminal types, each with different screen layouts, escape sequences for inserting and deleting text, moving the cursor, etc.

One way to solve this problem is to define an abstract network virtual terminal that editors and other programs can be written to deal with.

To handle each terminal type, a piece of software must be written to map the functions of the network virtual terminal onto the real terminal.

* TCP/IP Reference Model :- When satellite and radio networks were added later, the existing protocols had trouble interworking with them, so a new reference architecture was needed. Thus the ability to connect multiple networks together in a seamless way was one of the major design goals from the very beginning. This architecture later became known as the TCP/IP Reference Model.



TCP/IP.

① Host-to-network layer :- Below the internet layer is a great void. 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 over it. —

This protocol is not defined and varies from host-to-host and network-to-network

(ii) Internet layer :- All these requirements led to the choice of a packet-switching network based on a connectionless internetwork 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 them 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.

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.

(iii) The Transport layer :- It is designed to allow peer entities on the source and destination hosts to carry on a conversation, the same as in the OSI transport layer.

Two end-to-end protocols have been defined here.

(a) TCP [Transmission Control Protocol] :- It 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 onto 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.

(b) UDP [User Datagram Protocol] :- It 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.

(iv) Application layer: The TCP/IP model does not have session or presentation layers. No need for them was perceived, so they were not included. Experience with the OSI model has proven this view correct: they are of little use to most applications.

It contains all the higher-level protocols. The early ones included TELNET, FTP, SMTP (short message transfer protocol). Many other protocols have been added to these over the years, such as the DNS (Domain Name Service) for mapping host names onto their network addresses, NNTP, the protocol used for moving news articles around, and HTTP, the protocol used for fetching pages on the World Wide Web.

Unit-2

→ Data link layer :-

(i) ~~Services Provided~~

* Data link layer Design Issues :- 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) Dealing with transmission errors
(3) Regulating the flow of data so that slow receivers are not swamped by fast senders.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into frames for transmission.

(i) Services Provided to the Network layer :- The func^{n} of the data link layer is to provide services 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 data link layer can be designed to offer various services. The actual services offered can vary from system to system. Three reasonable possibilities that are commonly provided are :-

1) Unacknowledged connectionless service
2) Acknowledged connectionless service
3) Acknowledged connection-oriented service.

Unacknowledged connectionless service consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them.

The next step in terms of reliability is acknowledged connectionless service. When this service is offered, there are still no logical connections used, but each frame sent is individually acknowledged. In this way, the sender knows whether a frame has arrived correctly. If it has not arrived within a specified time interval, it can be sent again. This service is useful over unreliable channels, such as wireless systems.

The most sophisticated service the data link layer can provide to the network layer is 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 indeed received. Furthermore, it guarantees that each frame is received exactly once and that all frames are received in the right order.

Imp

* 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 a raw bit stream and attempt to deliver it to the destination. This bit stream is not guaranteed to be error free. The number of bits received may be less than, equal to, or more than the number of bits transmitted, and they may have different value. 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 a frame arrives at the destination, the checksum is recomputed. If the newly-computed checksum is different from the one contained

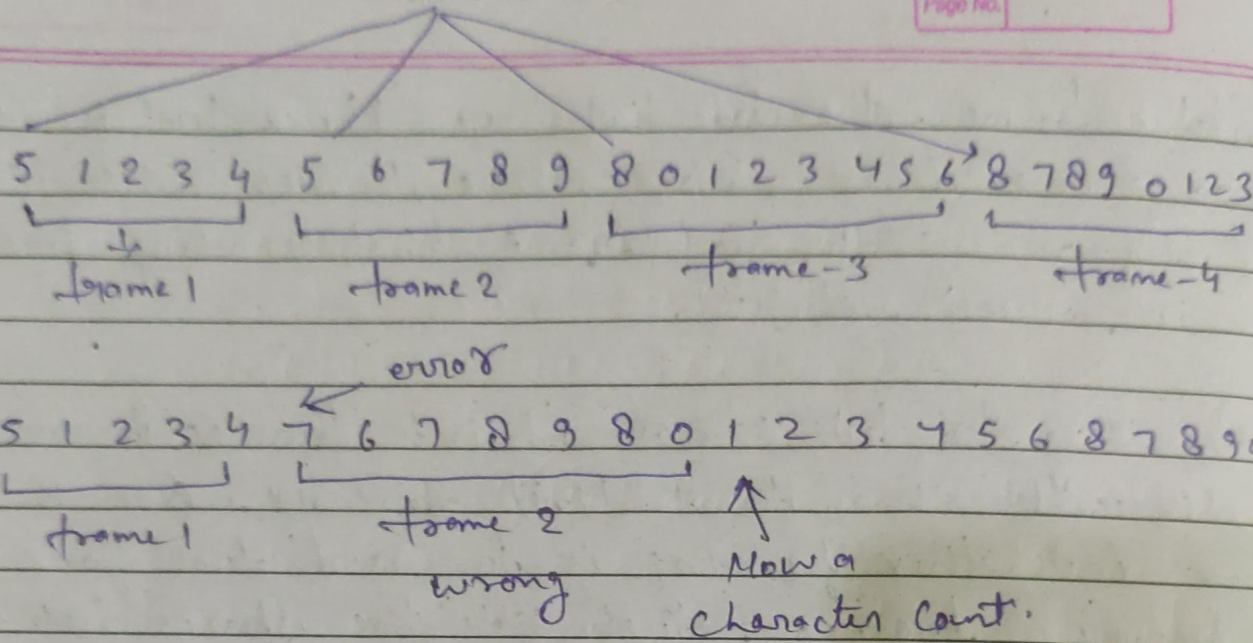
in the frame, the data link layer knows that an error has occurred and takes steps to deal with it (discarding the bad frame and sending back an error report).

Breaking the bit stream up into frames is more difficult than it at first appears. One way to achieve this framing is to insert time gaps between frames, much like the spaces between words in ordinary text. However, networks rarely make any guarantees about timing, so it is possible these gaps might be squeezed out or other gaps might be inserted during transmission.

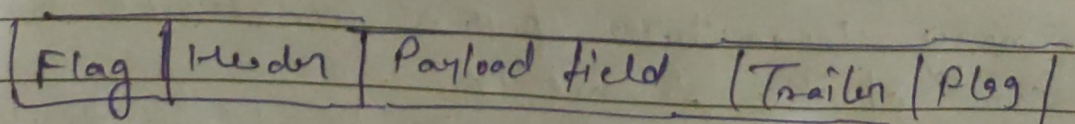
Since it is too risky to count on timing to mark the start and end of each frame, other methods have been devised.

① Character count :→ It 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 and hence where the end of the frame is.

The trouble with this algorithm is that the count can be garbled by a transmission error.



(2) Flag bytes with byte stuffing :- This method gets around the problem of resynchronization after an error by having each frame start and end with special bytes. In the past, the starting and ending bytes were different, but in recent years most protocols have used the same byte, called a flag byte, ~~or both the starting~~ as both the starting and ending delimeter. In this way, if the receiver ever loses synchronization, it can just search for the flag byte to find the end of the ~~new~~ current frame.



Tmp → replace byte with little

Date
Page No.

(3) Starting and ending flags, with bit stuffing :-

As networks developed, the disadvantages of embedding the character code length in the framing mechanism became more and more obvious, so a new technique had to be developed to allow arbitrary sized characters.

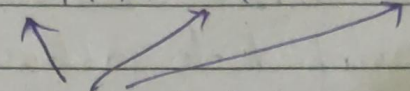
The new technique allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character. It works like this. Each frame begins and ends with a special bit pattern, 01111110 (in fact, a flag byte). Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the outgoing bit stream. This bit stuffing is analogous to byte stuffing, in which an escape byte is stuffed into the outgoing character stream before a flag byte in the data.

When the receiver sees five consecutive incoming 1's, followed by a 0 bit, it automatically destuffs the 0 bit. Just as byte character stuffing is completely transparent to the network layer in both computers, so is bit stuffing. If the user data contain the flag pattern, 01111110, this flag is transmitted as 01111010

but stored in the receiver's memory as 01111110.

a) 0110111111111111111110010

b) 01101111101111101111010010



With bit stuffing, the boundary between two frames can be unambiguously recognized by the flag pattern. Thus, if the receiver loses track of where it is, all it has to do is scan the input for flag sequences, since they can only occur at frame boundaries and never within the data.

* Error Control :- Having solved the problem of marking the start and end of each frame, we come to the next problem: how to make sure all frames are eventually delivered to the network layer at the destination and in the proper order.

* 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 the receiver can accept them. This situation can easily occur when the sender is running on a fast (or lightly loaded) computer and the receiver is running on a slow (or heavily loaded) machine. The sender keeps pumping the frames out at a high rate until the receiver is completely swamped.
Even if the transmission is error free, at a certain point the receiver will simply be unable to handle the frames as they arrive and will start to lose some.

The usual situation is to introduce flow control to throttle the sender into sending no faster than the receiver can handle the trap. → feedback mechanism.

* Error Detection and Correction :- In transmission, errors are basically backlog. An error has occured but not what the error is.

But in Error Correction it introduce enough redundant info. that the system is able to detect the error and also

able to know to what the error is.

* Error-Correcting Codes: Networks designers have developed two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been. The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission. The former strategy uses error-correcting codes and the latter uses error-detecting codes.

To understand how errors can be handled, it is necessary to look closely at what an error really is. Normally, a frame consists of m data (message bits) and r redundant or check bits. Let the total length be

$$n = m + r$$

An n-bit until containing data and check bits is often referred to as an n-bit codeword.

Given any two codewords, say, 10001001 and 10110001, it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just exclusive OR the two codewords and count the number of 1 bits in the result, for example:

```
10001001
10110001
--------
00111000
```

The number of bit positions in which two codewords differ is called the Hamming distance. Its significance is that if two codewords are a Hamming distance d apart, it will require d single-bit errors to convert one into the other.

In most data transmission applications, all 2^m possible data messages are legal, but due to the way the check bits are computed, not all of the 2^n possible codewords are used. Given the algorithm for computing the check bits, it is possible to construct a complete list of the legal codewords, and from this list find the two codewords whose

Hamming distance is minimum. This distance is the Hamming distance of the complete code.

The error-detecting and error-correcting properties of a code depend on its Hamming distance. To detect d errors, you need a distance d+1 code because with such a code there is no way that d single-bit errors can change a valid codeword into another valid codeword. When the receiver sees an invalid codeword, it can tell that a transmission error has occurred. Similarly, to correct d errors, you need a distance 2d+1 code because that way the legal codewords are so far apart that even with d changes, the original codeword is still closer than any other codeword, so it can be uniquely determined.

As a simple example of an error-detecting code, consider a code in which a single parity bit is appended to the data. The parity bit is chosen so that the number of 1 bits in the codeword is even (or odd). For example, when 1011010 is sent in even parity, a bit is added to the end to make it 10110100. With odd parity 1011010 becomes 10110101. A code with a single parity bit has a distance 2, since

any single-bit error produces a codeword with the wrong parity. It can be used to detect single errors.

As a simple example of an error-correcting code, consider a code with only four valid codewords:

0000000000, 0000011111, 1111100000, 1111111111

This code has a distance 5, which means that it can correct double errors. If the codeword 0000000111 arrives, the receiver knows that the original must have been 0000011111. If, however, a triple error changes 0000000000 into 0000000111, the error will not be corrected properly.

Imagine that we want to design a code with m message bits and r check bits that will allow all single errors to be corrected. Each of the 2^m legal messages has n illegal codewords at a distance 1 from it. These are formed by systematically inverting each of the n-bits in the n-bit codeword formed from it. Thus, each of the 2^m legal messages requires n+1 bit patterns dedicated to it. Since the total number of bit patterns is 2^n, we

we must have $(n+1)2^m \leq 2^n$.

Using $n = m + r$, this requirement becomes $(m+r+1) \leq 2^r$. Given m, this puts a lower limit on the number of check bits needed to correct single errors.

This theoretical lower limit can, in fact, be achieved using a method due to Hamming. The bits of the codeword are numbered consecutively, starting with bit 1 at the left end. The bits that are powers of 2 (1, 2, 4, 8, 16, etc) are check bits. The rest (3, 5, 6, 7, ---) are filled up with m data bits. Each check bit forces the parity of some collection of bits, including itself, to be even (or odd). A bit may be included in several parity computations. To see which check bits the data bit in position k contributes to, rewrite k as a sum of powers of 2. For example: $11 = 1 + 2 + 8$. A bit is checked by just those check bits occurring in its expansion.

When a codeword arrives, the receiver initializes a counter to zero. It then examines each check bit, k ($k = 1, 2, 4, 8, \ldots$) to see if it has the correct parity. If not, the receiver adds k to the counter. If the

counter is zero after all the check bits have been examined (i.e., if they were all correct), the code-word is accepted as valid. If the counter is nonzero, it contains the number of the incorrect bit. For example, if check bits 1, 2, and 8 are in error, the inverted bit is 11, because it is the only one checked by bits 1, 2, and 8.

* Error-Detecting Codes: The polynomial code, also known as CRC (Cyclic Redundancy Check) are based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only. A k-bit frame is regarded as the coefficient list for a polynomial with k terms, ranging from x^{k-1} to x^0.

Such a polynomial is said to be of degree k-1. The higher-order (leftmost) bit is the coefficient of x^{k-1}; the next bit is the coefficient of x^{k-2}, and so on. For example, 110001 has 6 bits and thus represents a six-term polynomial with coefficients 1, 1, 0, 0, 0 and 1.

Polynomial arithmetic is done modulo 2, according to the rules of algebraic field theory.

There are no carries for addition or borrows for subtraction. Both addition and subtraction are identical to exclusive OR.

Long division is carried out the same way as it is in binary except that the subtraction is done modulo 2, ~~as above~~. A divisor is said "to go into" a dividend if the dividend has as many bits as the divisor.

When the polynomial code method is employed, the sender and receiver must agree upon a generator polynomial, $G(x)$, in advance. Both the high and low order bits of the generator must be 1. To compute the checksum for some frame with m bits, corresponding to the polynomial $M(x)$, the frame must be longer than the generator polynomial. The idea is to append a checksum to the end of the frame in such a way that the polynomial represented by the checksummed frame is divisible by $G(x)$. When the receiver gets the checksummed frame, it tries dividing it by $G(x)$. If there is a remainder, there has been a transmission error.

The algorithm for computing the checksum is as follows :—

1) Let r be the degree of G(x). Append r zero bits to the low-order end of the frame so it now contains m+r bits and corresponds to the polynomial $x^r M(x)$.

2) Divide the bit string corresponding to G(x) into the bit string corresponding to $x^r M(x)$, using modulo 2 division.

3) Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to $x^r M(x)$ using modulo 2 subtraction.

The result is the checksummed frame to be transmitted. Call its polynomial T(x).

Frame : 1101011011

Generator : 10011

Message after 4 zero bits are appended:

11010110110000

Remainder :- 1110

Transmited frame :- 11010110111110