Overview: Network Basics

- **types of communication service**
- **how communication services are implemented**
  - **switching**
  - **multiplexing**
- **network performance measures**

Types of service in a layered network architecture

- **connection-oriented:**
  - ➢ establish a connection
  - ➢ use the connection (for data transfer)
  - ➢ release the connection
  - modelled on the telephone system
  - essential feature: sender pushes objects (e.g. bits, packets) in at one end of the connection, and receiver takes them out ***in the same order*** at the other end

- **connectionless:** each message is sent independently of any other messages going from the same sender to the same receiver
  - modelled on the postal service
  - essential features: each message must ***include the receiver's address***, and messages can be received in a ***different order*** to the order in which they were transmitted

Types of service in a layered network architecture (cont.)

- each type of service can be characterised by its **reliability:** whether or not the service guarantees to correctly deliver the data
    - a reliable service is typically implemented by having the receiver confirm to the sender that it correctly received each message (which introduces extra overhead and delays)

- *reliable connection-oriented service* has 2 variations:
    - <u>message stream:</u> preserves message boundaries (e.g. 2 1-kB messages are received as 2 1-kB messages, **not** 1 2-kB message, 4 512-byte messages, or anything else)
    - <u>byte stream:</u> doesn’t preserve message boundaries (e.g. 2 1-kB messages are received as a 2048-byte stream)
- can also have *unreliable connection-oriented service*
    - e.g. real-time audio or video: tolerates some errors or losses in transmission (quality decreases as errors/losses increase)

Types of service in a layered network architecture (cont.)

• *connectionless service* can be unreliable (no 100% delivery guarantee), acknowledged (receipt confirmed), or request-reply (a single short message contains a request, another the reply)

	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
	Unreliable connection	Digitized voice
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

Types of service in a layered network architecture (cont.)

- **Note:** communication services are defined based on end-to-end properties, ***not*** on how bits are transported in the network
  - a single network could offer *more than one type of service*
    - e.g. the Internet supports all of them, more or less
  - may have *different implementations* of the same service
    - e.g. connection-oriented delivery of a voice bit stream can be implemented by <u>packet voice</u> or a <u>dedicated circuit</u>

How are communication services implemented ?

• it is economically infeasible to directly connect every pair of sender-receiver pairs (e.g. mesh or star) in a large network

• technology limitations mean that broadcast solutions don't scale to large numbers of hosts or large geographical distances

• therefore network resources must be ***shared*** between the users, while still allowing senders to transmit data to their receivers

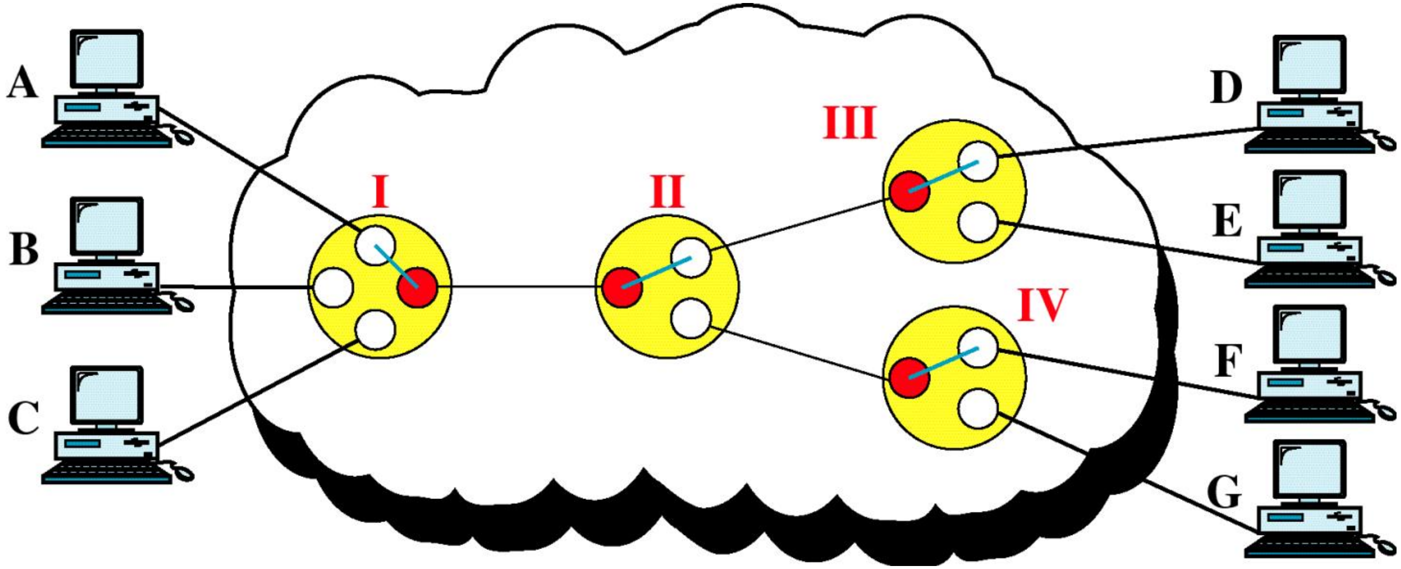
• the two basic techniques that permit connectivity while sharing resources are **switching** and **multiplexing**

- ➢ switching: sharing *network resources* among multiple transmissions
- ➢ multiplexing: sharing a *single link* among multiple transmissions

Circuit switching

• a **path** is set up in the network between the sender and the receiver (by making the appropriate connections in the switches)

- the necessary network resources are **reserved** for the connection prior to any data transfer; if this is not possible, the connection request is **blocked**
- these reserved resources are then **held** for the duration of the connection, regardless of actual usage



Switch = a device that can create a temporary connection between an input link and an output link

Circuit switching (cont.)

- network links are *not shared at the same time*
  - the links in a path are monopolised for the duration of the connection, then released so that they are available for other connections

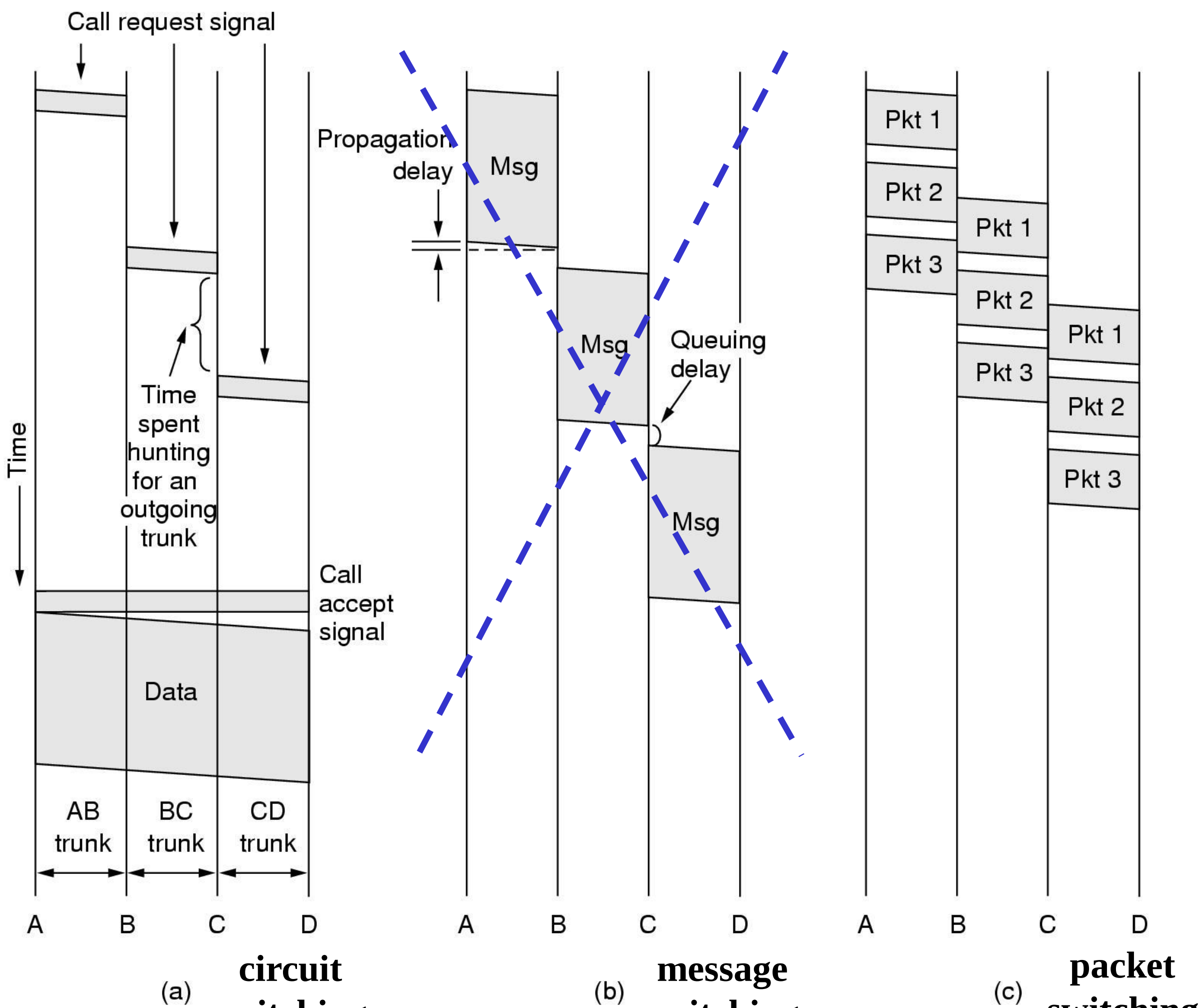
- the connection set-up delay can be significant (>1 second)

- circuit switching is ideal for “smooth” network traffic
  - e.g. telephone network

- what if the traffic from sender to receiver is “bursty” (which means it varies widely around its average value) ?
  - computer-to-computer traffic can be very bursty
    - ➢ could set up a new circuit for each burst
    - ➢ could hold original circuit for duration of data transfer
  - both of these solutions are wasteful of network resources

Packet switching

- packet = string of bits (up to a few thousand, typically)
- uses **store-and-forward** operation:



circuit switching **message switching** **packet switching**

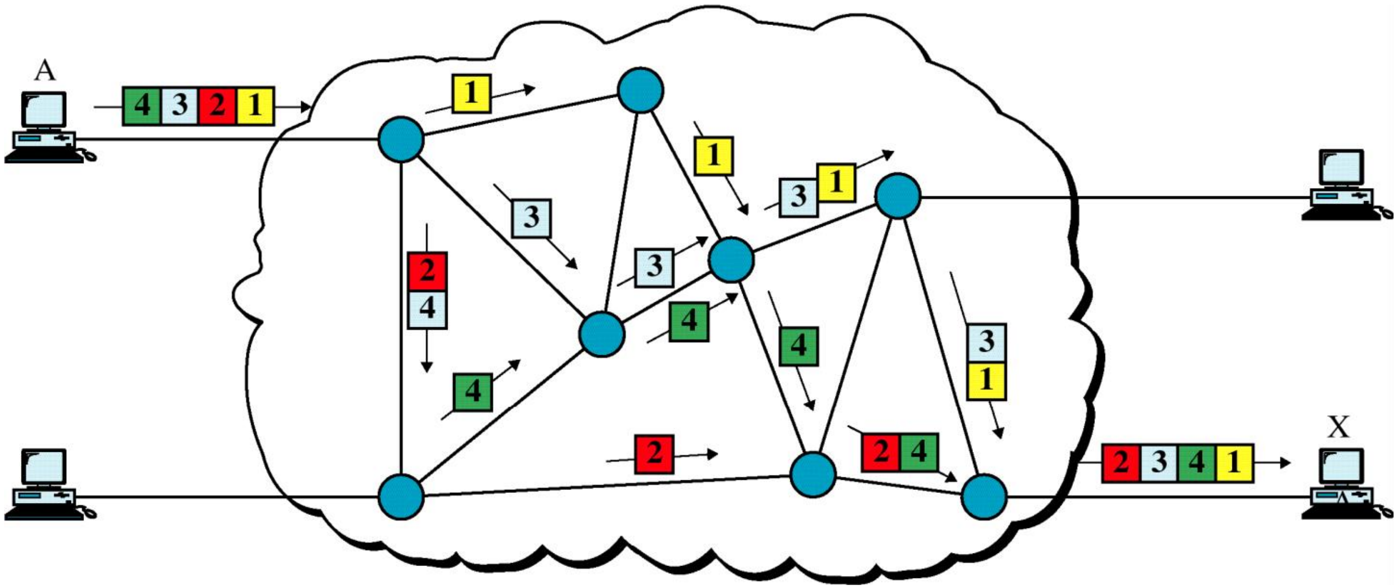
In packet switching, an <u>upper limit</u> is placed on packet size, and packet transmissions are <u>pipelined</u> (reducing the overall delay)

Packet switching (cont.)

• 2 basic types of packet switching: datagram vs. virtual circuit

• **datagram packet switching:** each packet is treated individually within the network, so successive packets may follow different routes through the network

  • each packet contains the receiver's address and a sequence number (so that receiver can put them into correct order)



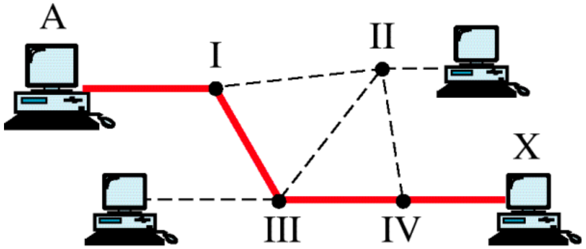
Network nodes are routers, which have <u>routing tables</u> telling them which output link to use for each possible destination

Packet switching (cont.): Datagram packet switching

- no connection set-up needed
- flexible routing possible (e.g. if a router crashes)
- network resources are *not shared at the same time*
    - each packet monopolises a link during its transmission, after which the link is available for other packet transmissions
- datagram p.s. ideal for *short-lived* bursty traffic
- datagram p.s. less suitable for *long-lived* &/or *interactive* bursty traffic

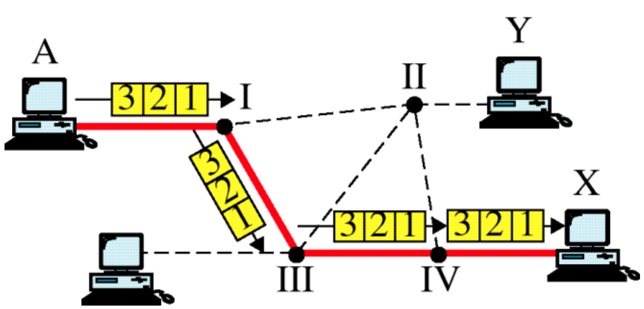
Packet switching (cont.)

- **Virtual circuit packet switching:** a **route** is set up in the network between sender and receiver (by making appropriate entries in the routing tables)
    - resources may or may not be reserved for this route. If resources need to be reserved and are not available, the connection request is **blocked**
    - each packet contains its virtual circuit identifier

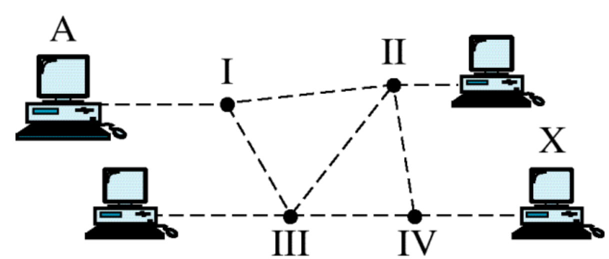


Connection establishment

Routers have <u>routing tables</u> telling them which output link to use for each established virtual circuit



Data transfer



Connection release

Packet switching (cont.): Virtual circuit packet switching

- connection set-up required, which may involve significant delay

- network resources are *not shared at the same time*
    - each packet monopolises a link during its transmission, after which the link is available for other packet transmissions

- less work required at intermediate routers than for datagram p.s.
    - given a packet's input link and virtual circuit identifier, the router can look up its routing table to find the output link

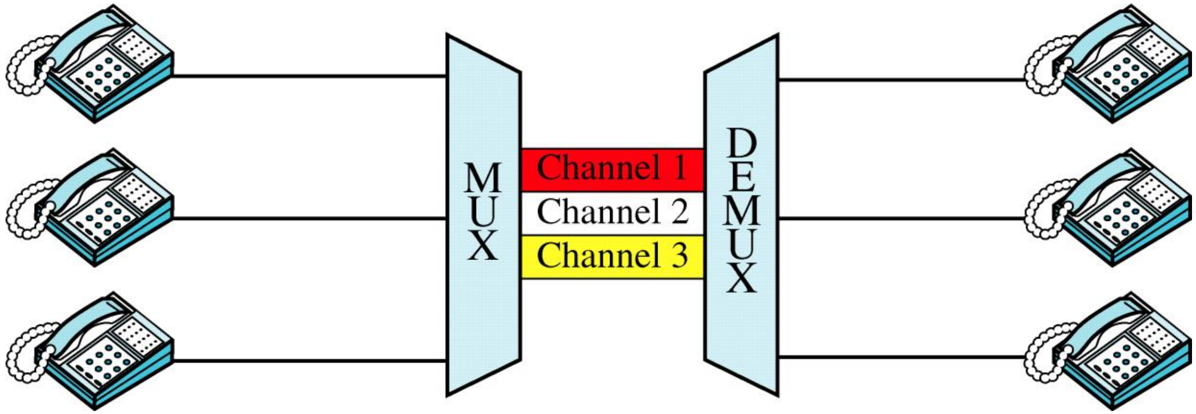
- virtual circuits not as robust to network problems as datagram p.s.

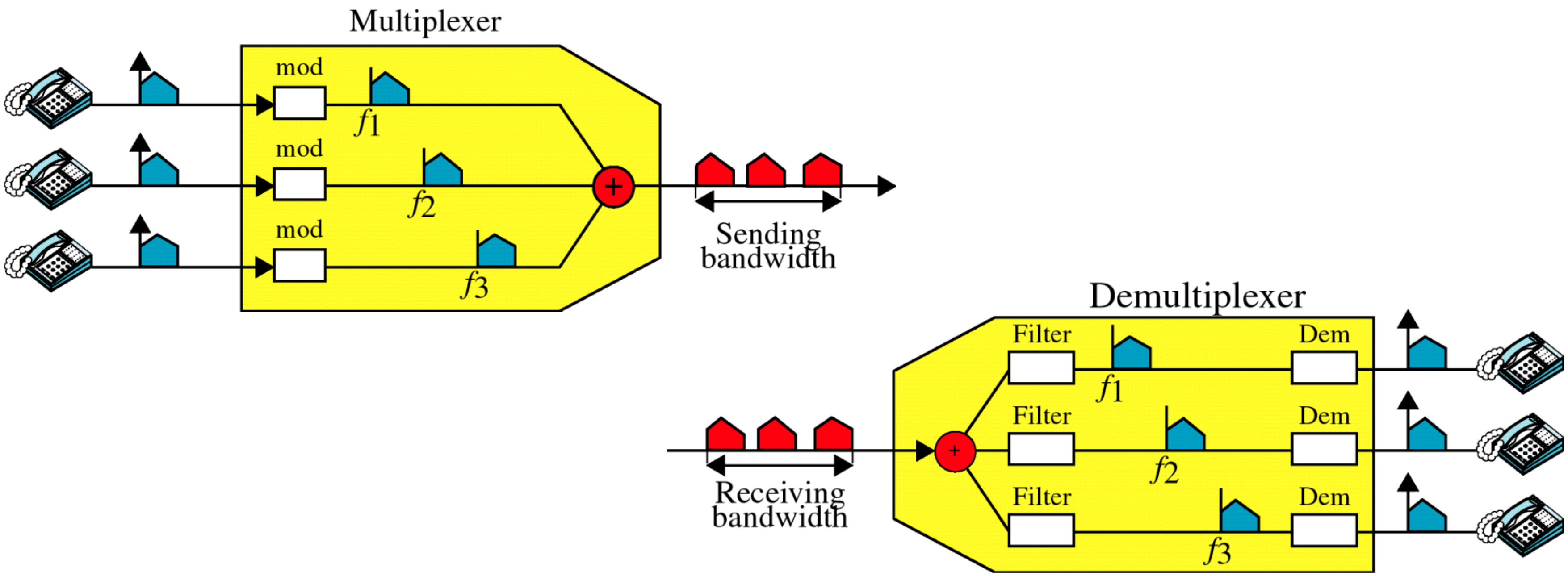
- virtual circuit p.s. represents a "compromise" between circuit switching and datagram p.s.
    - circuit switching creates a <u>path</u> in the network; virtual circuit p.s. creates a <u>route</u> which exists only in software; datagram p.s. doesn't have routes
    - in circuit switching, the links in the path <u>cannot be shared</u> during the connection; in virtual circuit and datagram p.s. they <u>can</u>

Multiplexing

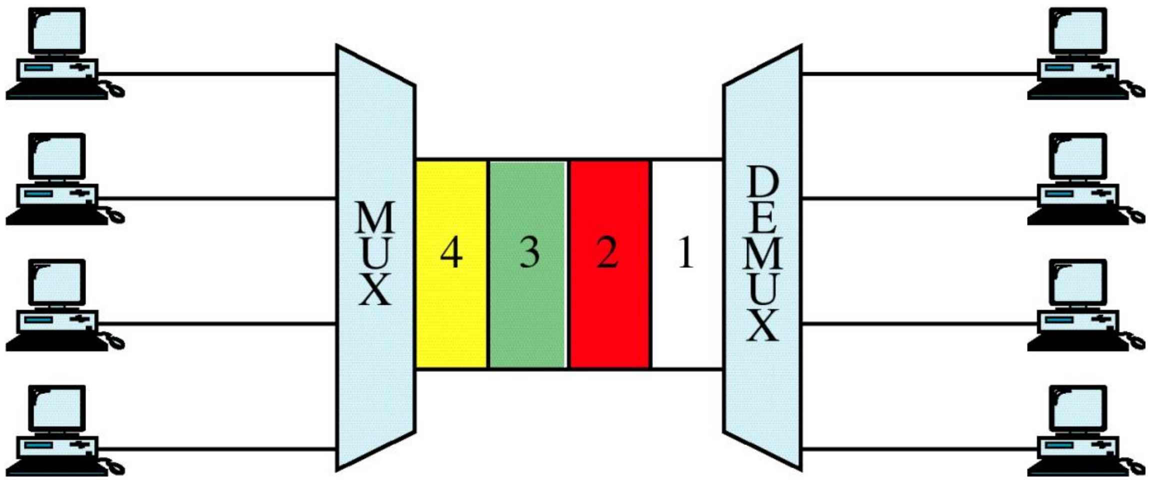
- sharing a single link among multiple transmissions
- 3 basic possibilities:
    - Frequency division multiplexing (FDM)
    - Time division multiplexing (TDM)
    - Statistical multiplexing

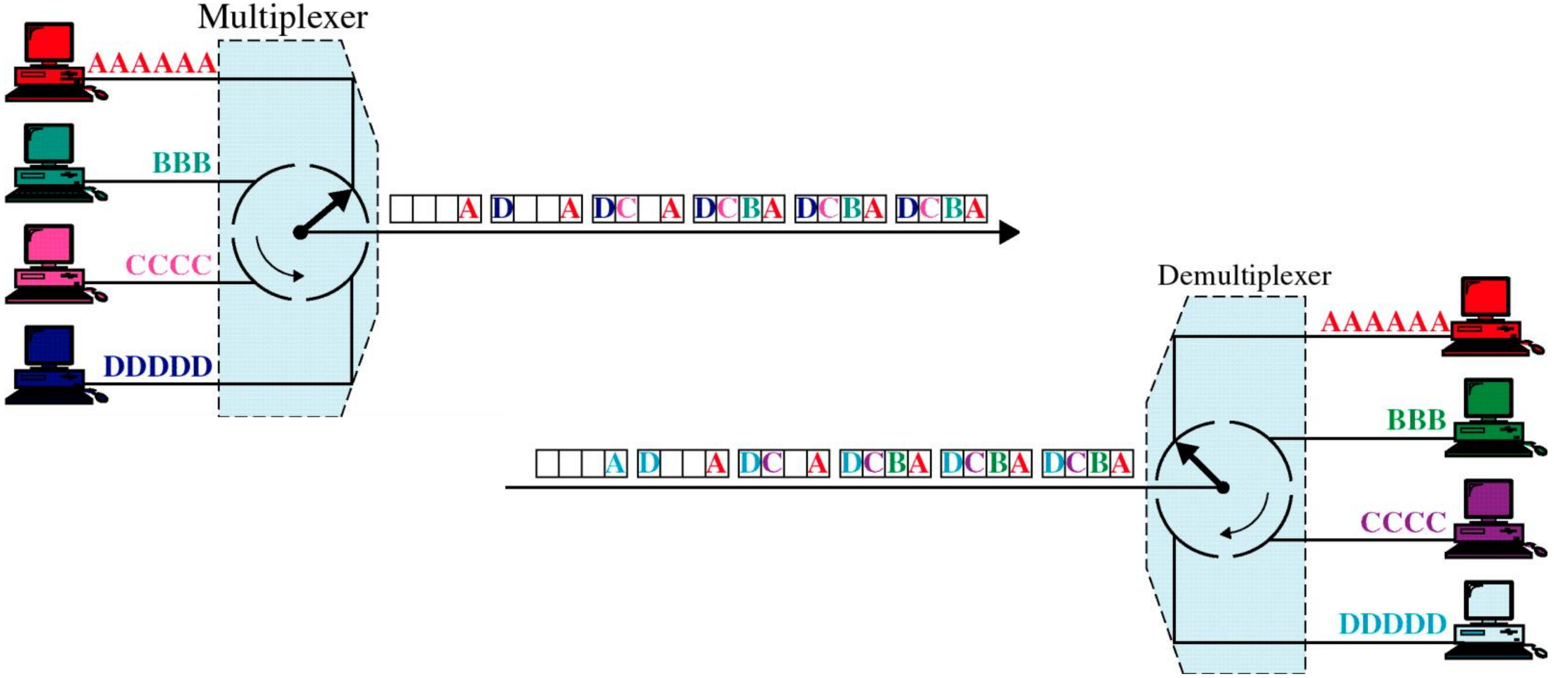
Multiplexing: FDM





Multiplexing: TDM

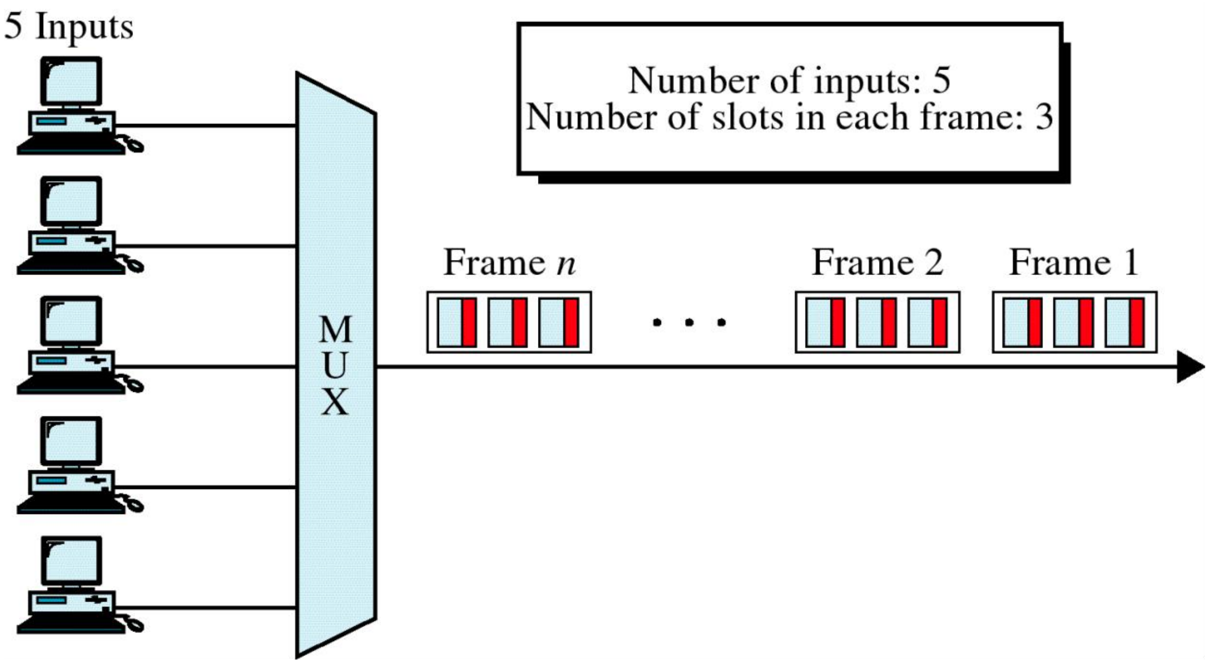




Multiplexing: Statistical multiplexing

- both FDM and TDM divide the link into **independent** channels
  - inefficient if traffic is bursty, since no sharing allowed

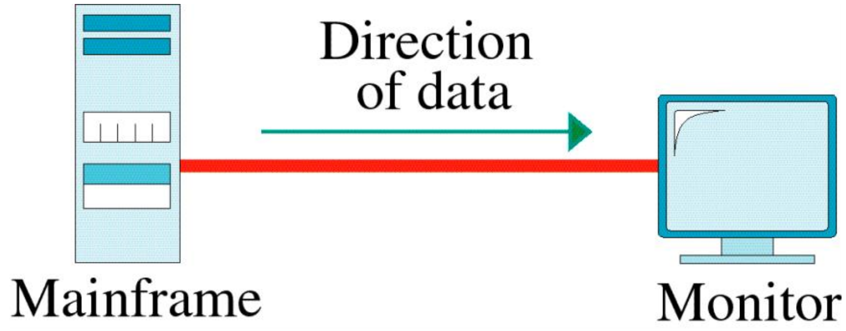
- in statistical multiplexing, the idea is that the link should never be idle when there is data to be transferred



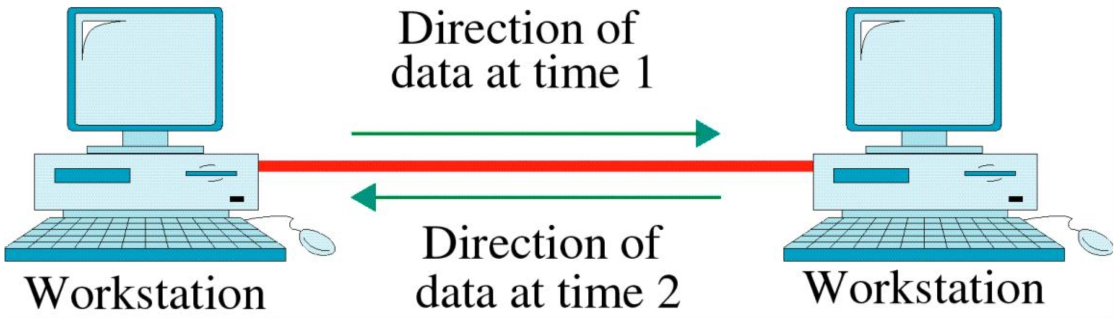
Frames have an additional overhead (compared to FDM or TDM) to indicate which input stream they belong to; the packet queueing delay is now variable; and a (possibly complex) method is needed to decide which packets to multiplex

Transmission modes

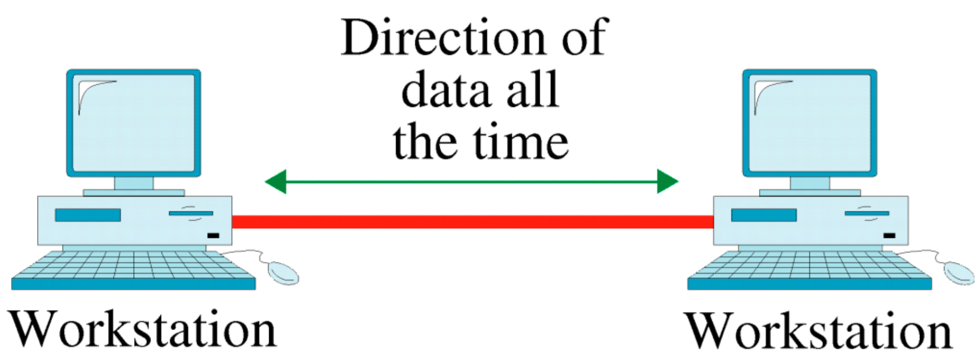
Simplex



Half-duplex



Full-duplex



Network performance measures

l = length of signal path in communication medium (metres)
v = signal propagation speed in the medium (metres/second)
L = average length of frame or packet (bits)
C = transmission rate (bits/second)

- **Propagation delay** = l / v , in seconds
  - shows how long a bit takes to propagate along the path
- **Transmission time** = L / C , in seconds
  - shows how long it takes to get packet onto the medium
- **Throughput:** how fast data can pass a certain point
  - can be measured in bits/second, packets/second, …
- **Efficiency** is related to throughput, e.g.

efficiency = throughput (in packets/sec) * packet transmission time

Network performance measures: Example 1

Consider an optical fibre 3000 km long with a transmitter transmitting at 1.5 Gbps (1 Gbps = 1 000 000 000 bps). The signal propagation speed in optical fibre is approximately 200 000 km/sec. Suppose packet switching is being used with a packet length of 2000 bits.

What is the bit propagation delay along the fibre ?

prop_delay = (3 000 000)/(200 000 000) = 15 millisec

What is the packet transmission time here ?

pkt_trans = (2 000)/(1 500 000 000) = 1.3333 microsec

How many packets have been transmitted and are propagating over the fibre when the first bit reaches the destination ?

num_pkts = $(15\times10^{-3})/(1.3333\times10^{-6})$ = 11 250 packets
(Note that this is 22 500 000 bits)

Network performance measures: Example 2

Consider a route in a store-and-forward network going through 8 intermediate nodes. The packets contain 1000 bits and are transmitted at 64 kbps. Assume propagation delays over the links are negligible. As a packet travels along the route, it encounters an average of 5 packets when it arrives at each node. How long does it take for the packet to get to the receiver if the nodes transmit on a "first come first served" basis ?

At each intermediate node, 6 packets must be transmitted in order for "our" packet to be transmitted: our packet finds 5 packets ahead of it, which will be transmitted first due to the "first come first served" policy.

The packet transmission time at every node is

$$pkt\_trans = (1\ 000)/(64\ 000) = 15.625 \text{ millisec}$$

The total travel time for our packet through the network is

$$\begin{aligned} travel\_time &= (\text{transmission delay at sender}) + \\ &\quad 8 \times (\text{delay at each intermediate node}) \\ &= pkt\_trans + 8 \times 6 \times pkt\_trans = 766 \text{ millisec} \end{aligned}$$

Network performance measures: Example 2 (cont.)

Note that the "pure" transmission delays only account for

route_trans = pkt_trans + 8 × pkt_trans = <u>141 millisec</u>

Our packet endures a queueing delay in each intermediate node of

node_q_delay = 5 × pkt_trans = 78 millisec

and since there are 8 intermediate nodes, the total queueing delay along this route is

route_q_delay = 8 × node_q_delay = <u>625 millisec</u>

which represents just over 80% of the total travel time.

So queueing delay can account for a substantial "extra" delay experienced by packets in the network. One way to reduce this queueing delay FOR SOME PACKETS would be to use a different policy in the intermediate nodes, rather than first-come-first-served. This could result in some "higher-priority" packets getting to their receivers much quicker, while "lower-priority" packets would experience longer delays.

Network performance measures: Example 3

150 nodes are connected to a 1000 metre length of coaxial cable. Using some (unspecified) protocol, each node can transmit 70 frames/second, where each frame is 1000 bits long. The transmission rate at each node is 100 Mbps.

What is the per-node throughput ?

$thru\_node = 70 \times 1000 = 70\ 000\ bps$

What is the total throughput (of the 150 nodes) ?

$thru\_total = 150 \times thru\_node = 10\ 500\ 000\ bps = 10.5\ Mbps$

What is the efficiency of this protocol ?

$$efficiency = (total\ throughput,\ in\ bps) \times (bit\ transmission\ time)$$

$$= (10\ 500\ 000) \times (1/100\ 000\ 000) = 0.105,\ or\ 10.5\%$$

$$OR\ efficiency = (total\ throughput,\ in\ frames\ per\ second) \times (frame\ transmission\ time)$$

$$= (70 \times 150) \times (1\ 000/100\ 000\ 000) = 0.105,\ or\ 10.5\%$$

Network performance measures: Example 3 (cont.)

What would give us 100% efficiency, and why is the efficiency so far below 100% here ?

If <u>some</u> node in the network was transmitting at every instant of time, the total throughput <u>would</u> be 100 Mbps and the efficiency <u>would</u> be 100%. However, the protocol being used to regulate access to the medium (the coaxial cable) introduces some delays between transmissions – either because it permits collisions to occur, which must be recovered from; or because some "permission to transmit" token must be passed from the node currently in possession to the next node allowed by the protocol to transmit. In either case, these inter-transmission gaps result in a drop in efficiency.

Note that, although this efficiency seems low, it may be perfectly acceptable. For example, if the objective was to run the network with a minimum throughput of 10 Mbps, this protocol works ok.