What is traveling on the wires?

Mixed data:

bulk data, audio/voice, video/image, real-time interactive data, etc.

⟶ > 85% of Internet traffic is bulk TCP traffic

⟶ due to Web and http

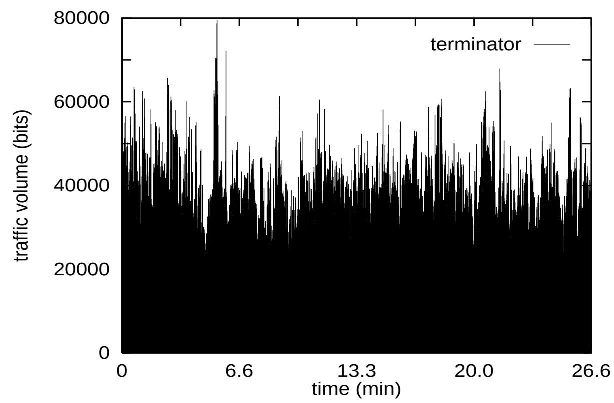
⟶ barriers to streaming traffic implosion

⟶ technical and other

Tilting toward *multimedia* data; i.e., traffic with QoS requirements including real-time constraints. Albeit slowly.

Internet traffic is bursty:

⟶ multimedia: MPEG compressed video



⟶ bulk data: 80/20 rule-of-thumb

⟶ majority of traffic is small, a few very large

⟶ "elephants and mice"

Data networks capable of carrying all types of data at the same time is a recent phenomenon.

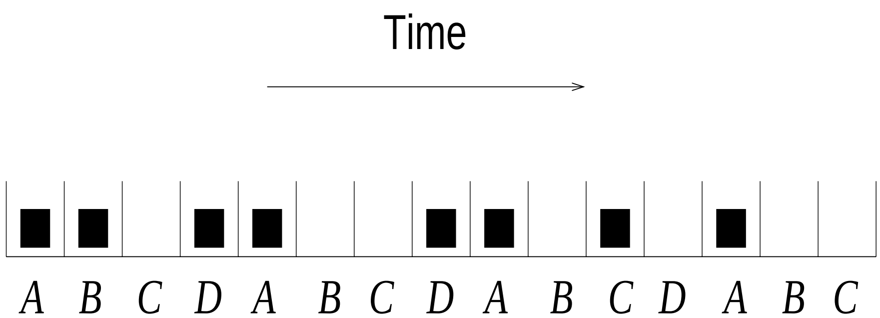
Even today, much of voice traffic (telephony) is carried on an entirely separate communication network vis-à-vis data traffic, operating under different internetworking principles from the latter.

$\longrightarrow$ time-division multiplexing (TDM) for telephony

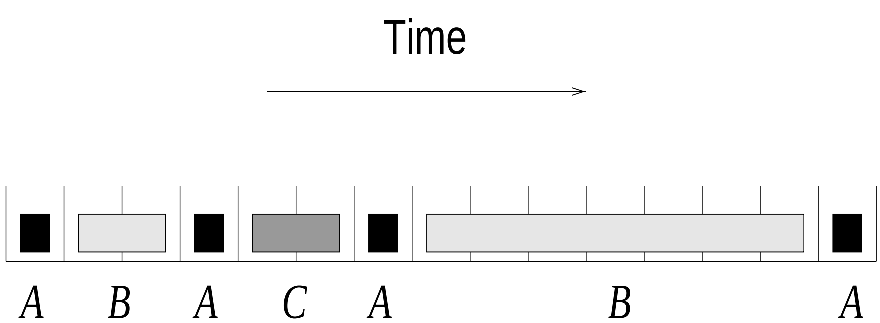
$\longrightarrow$ packet switching for data networks

How is "time" (viewed as a resource) shared?

Time-division multiplexing:

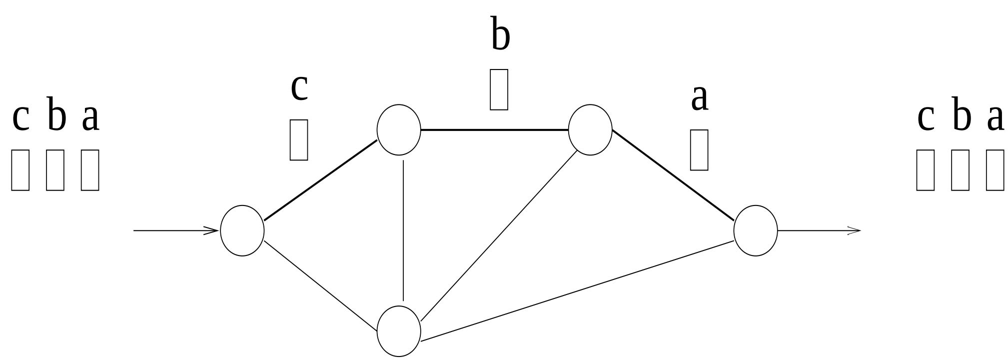


Packet switching:



How is "real estate" shared?

Circuit switching: Virtual channel is established and followed during the duration of an end-to-end conversation or connection.

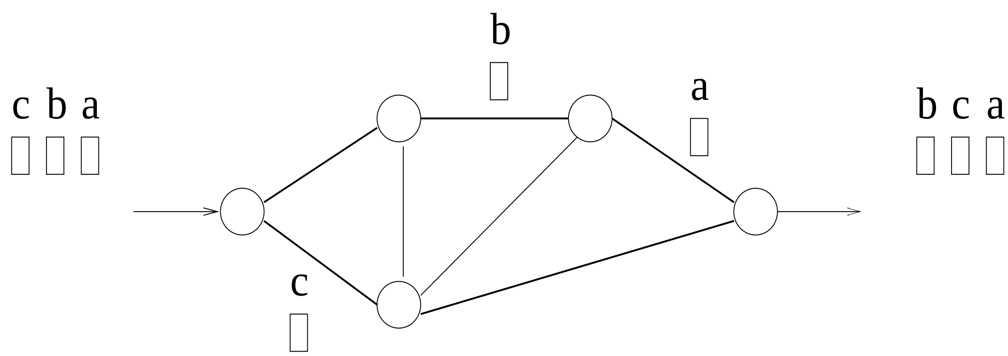


⟶ static route

⟶ in-order delivery

Telephone networks and ATM networks.

Packet switching: Every packet belonging to an end-to-end conversation or connection is an independent entity and may take a different route from other packets in the same conversation.



⟶ dynamic route

⟶ out-of-order delivery

Trade-off between processing overhead and route overhead; if route is fixed, the probability of the virtual circuit becoming an inefficient route increases with duration of conversation.

Trend: convergence to packet-switched technology

Immediate impediment:

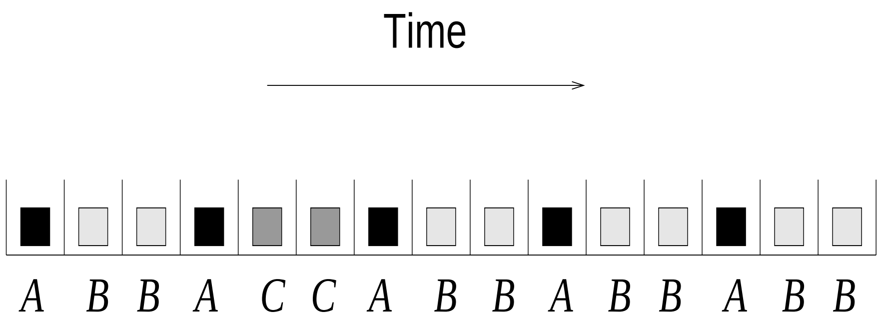
$\longrightarrow$ "bully phenomenon"

$\longrightarrow$ video: 24 frames-per-second (f/s)

$\longrightarrow$ voice: 8000 samples-per-second (s/s)

$\longrightarrow$ what to do?

Asynchronous transfer mode (ATM) :



⟶ 53 byte packet or *cell*.

Synergy of all forms of data, audio, video, bulk, etc. One unified network with "integrated" services.

⟶ addresses bully problem but ...

⟶ significant overhead (48 + 5)

⟶ duplicates forwarding functionality of IP

⟶ complex

How to make sense of all this?

Study of networks can be divided into three aspects:

- architecture
- algorithms
- implementation

Architecture

- hardware
  - communication or data link technology (e.g., Ethernet, SONET, CDMA/DSSS, TDMA)
  - hardware interface standards (e.g., EIA RS 232C—serial communication between DTE and DCE)
- software
  - conceptual organization (e.g., ISO/OSI 7 layer reference model, ATM network model)
  - protocol standards (e.g., IAB RFC—TCP, UDP, IP, Mobile IP; ISO MPEG)

$\longrightarrow$ the *what* over *how*

Provides the "shell" or "skeleton" for everything else.

... speaking of *standards*,

- ISO (International Standards Organization). ISO/OSI 7-layer reference model.
- ANSI (American National Standards Institute). U.S. Government representative at ISO.
- CCITT (International Telegraph and Telephone Consultative Committee). U.N.-chartered organization, principal international telecommunication standards organization.
- ITU (International Telecommunications Union). Successor of CCITT (used to be parent organization), U.N.-chartered.
- ATM Forum. Industry organization.
- IEEE. Professional society, LAN standards; e.g., IEEE 802.3 (Ethernet), IEEE 802.5 (token ring).
- IAB (Internet Architecture Board). Informal committee reporting to DoD, NSF.

- IETF (Internet Engineering Task Force). Subsidiary of IAB; main concern—short-term issues.
- IERF (Internet Research Task Force). IAB subsidiary; long-term issues.
- Internet Society. Elected trustees appoint IAB members.

Algorithms

- error detection and correction (e.g., checksum, CRC)
- medium access control (e.g., CSMA/CD, token ring, CSMA/CA)
- routing (e.g., shortest paths—Dijkstra, Bellman & Ford)
- congestion control or flow control (e.g., TCP window control and RTT estimation)
- scheduling (e.g., FCFS, weighted fair queueing)
- traffic shaping (e.g., leaky bucket) and admission control
- packet filtering (e.g., firewalls)
- overlay networks (e.g., VPNs)

$\longrightarrow$ *how* aspect of computer networks

Impacts network performance by controlling the underlying resources provided by the network architecture.

Implementations

Same algorithm can be implemented in different ways.

Key issue: *efficiency*.

- reduce copying operation

  $\longrightarrow$ pass pointers instead of value

- locality of reference

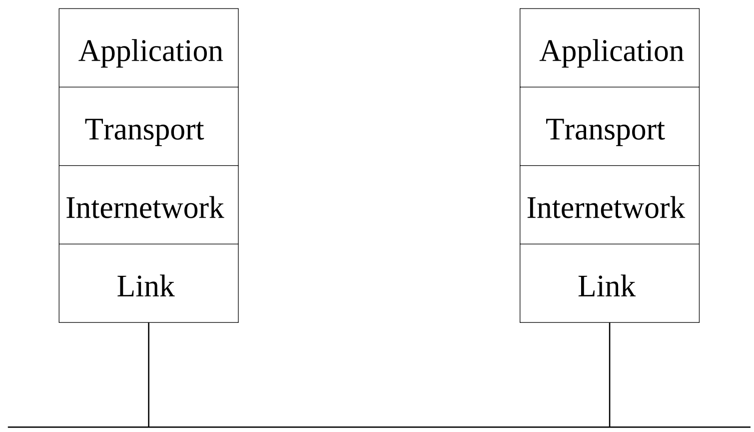
  $\longrightarrow$ packet trains

- header prediction
- multi-threading to reduce context-switch time overhead
- horizontal vs. vertical organization
- multi-threading to hide communication latency
- OS-support issues

Although at times ugly, a *must* to squeeze the best out of performance.

Architecture: Layering

Most fundamental organization of the software aspect of computer networks.



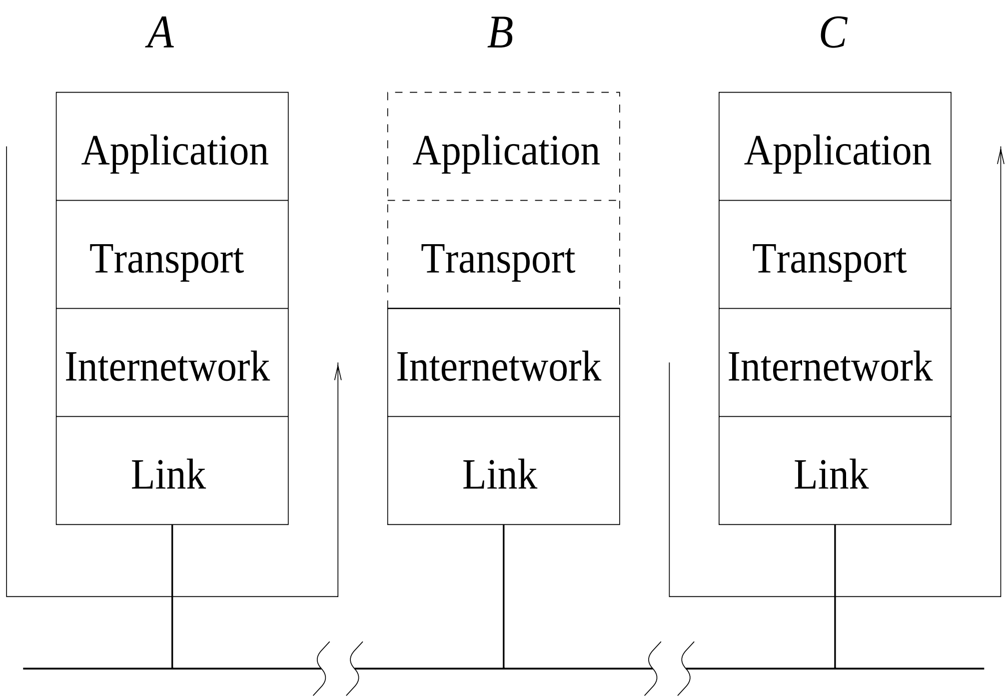
Achieves abstraction, modularization; two types of interfaces:

- peer-to-peer communication
- inter-layer communication (interface between layers)
  - SAP (service access point)
  - PDU (protocol data unit)

Functionalities:

- application layer—ftp, ssh, web browsers, etc.
- transport layer—end-to-end communication control (e.g., reliability, congestion control)
- internetwork layer—point-to-point control (e.g., routing, flow control)
- link layer—LAN access control

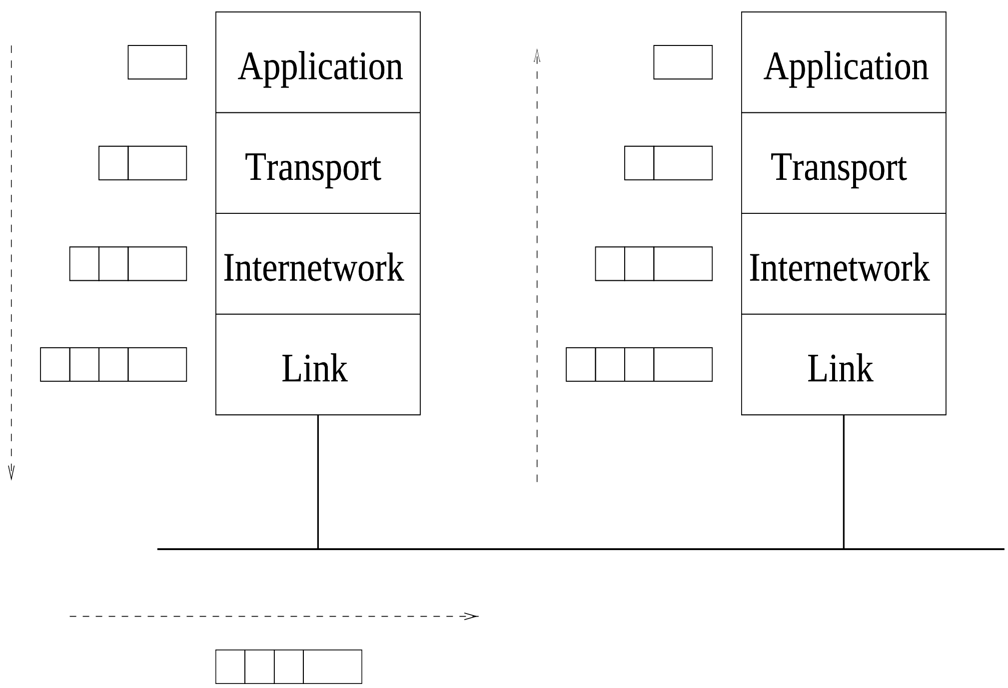
Internetworking example:



- application layers of A & C
- transport layers of A & C
- network layers of A & B and B & C
- link layers of A & B and B & C

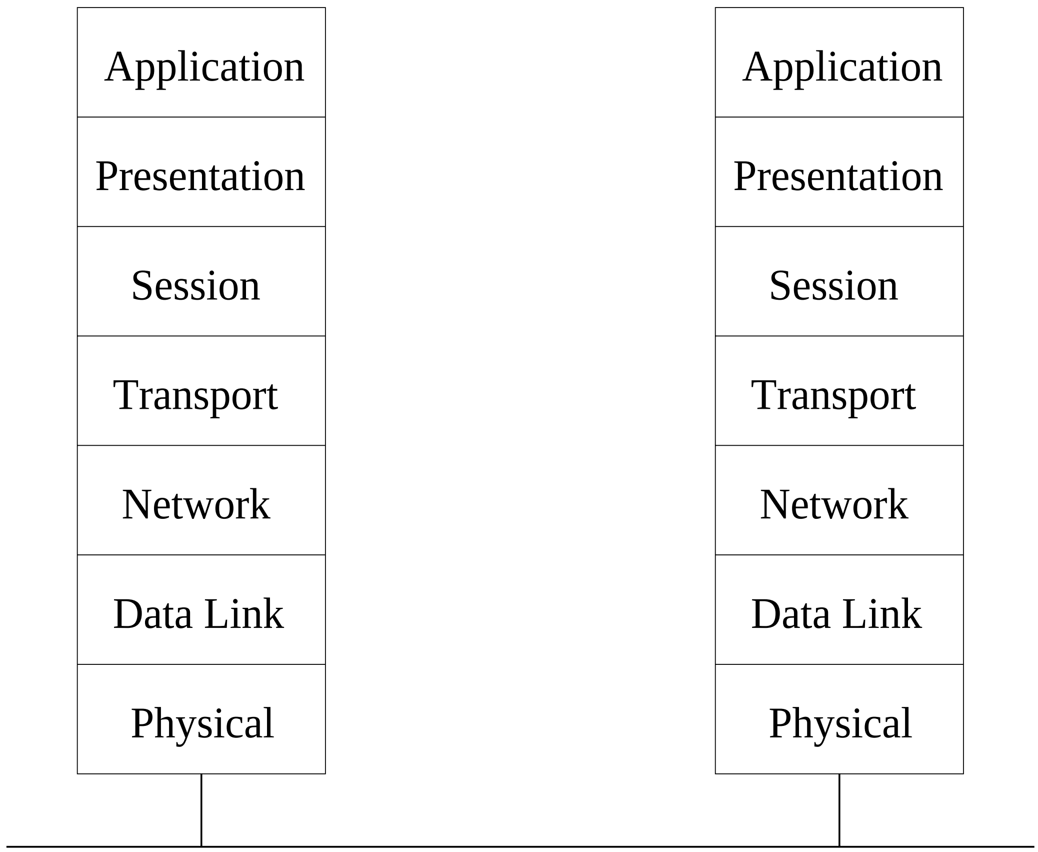
$\longrightarrow$ LAN switch/layer 2 switch: stops at link layer

Encapsulation:



- protocol stack (push/pop)
- header/trailer overhead
- segmentation/reassembly
- error detection etc.

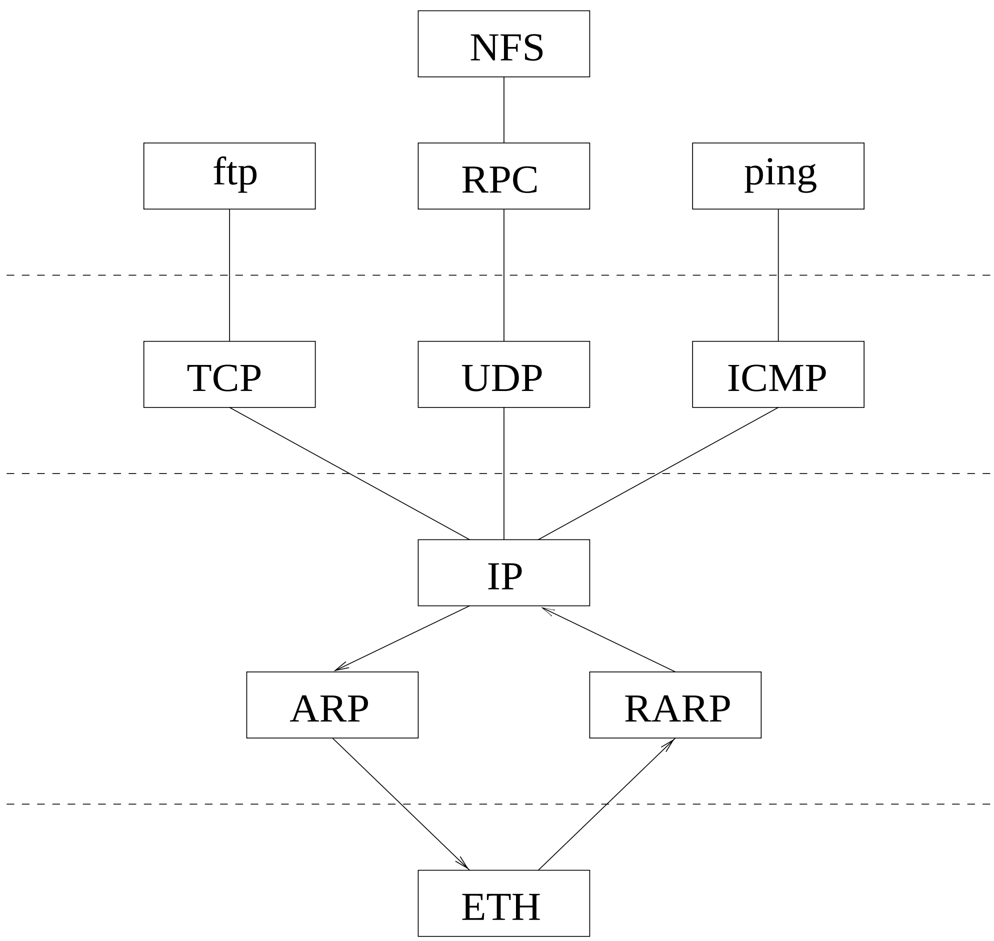
ISO/OSI 7-layer reference model:



Outdated; however, still useful as logical reference point.

Protocol graph:

Shows logical relationship between protocol modules in the protocol stack.



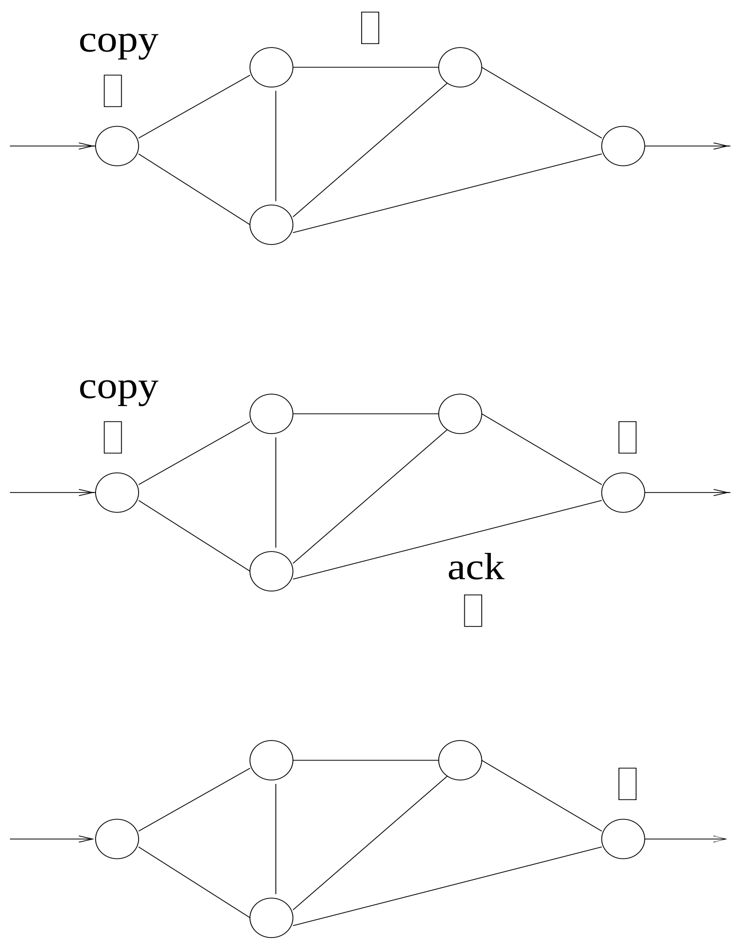
Algorithms: Reliable Communication

Packets may get

- corrupted due to errors (e.g., noise)
- dropped due to buffer overflow
- dropped due to aging or outdatedness—TTL (time-to-live field in IP)
- lost due to link or host failures

Internet philosophy: reliable transport (TCP) over unreliable internetwork (IP). Use retransmission and acknowledgment (ack).

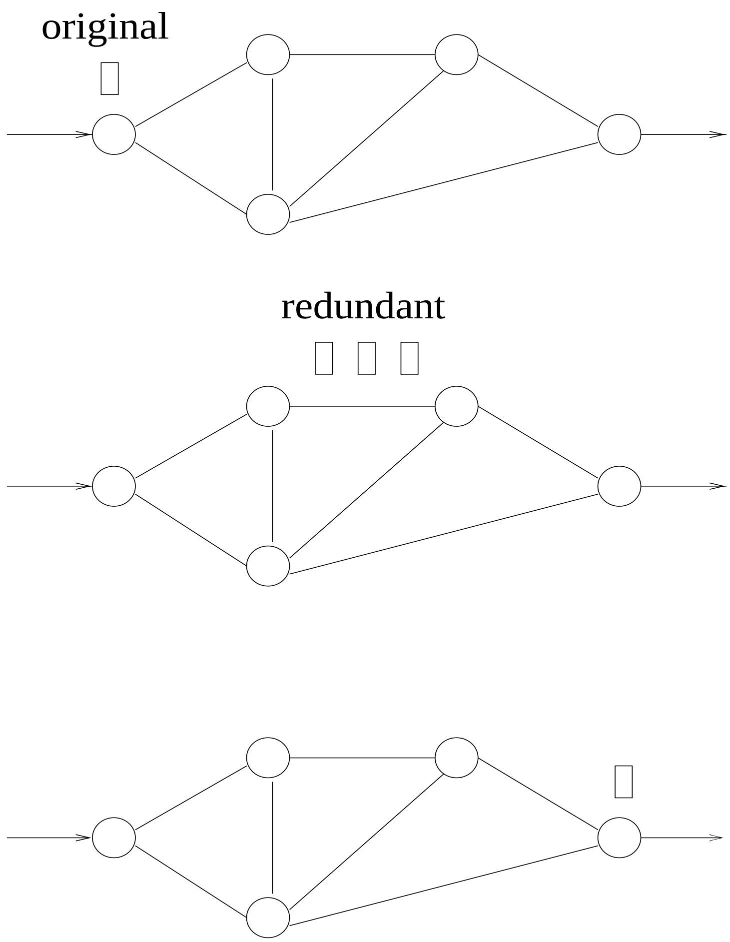
$\longrightarrow$ Automatic Repeat reQuest (ARQ)



- acknowledge receipt (positive ACK)
- absence of ACK indicates probable loss

... or vice versa (negative ACK scheme); when to use which.

Forward error-correction (FEC):



... works if at most two out of every three packets get dropped.

- send redundant information
- need to know properties of loss probability
- appropriate for real-time contrained data

$\longrightarrow$ FEC vs. "BEC" (backward error-correction)

Main drawback vis-à-vis ARQ?

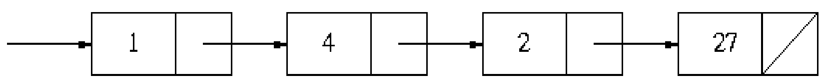
Implementation: Optimized Performance

Software clock:

⟶ single hardware clock to emulate multiple clocks

⟶ timer for keeping track of events

Example: want to be notified at time 1 sec, 5 sec, 7 sec, 34 sec from now.



Hardware clock interrupt handling routine:

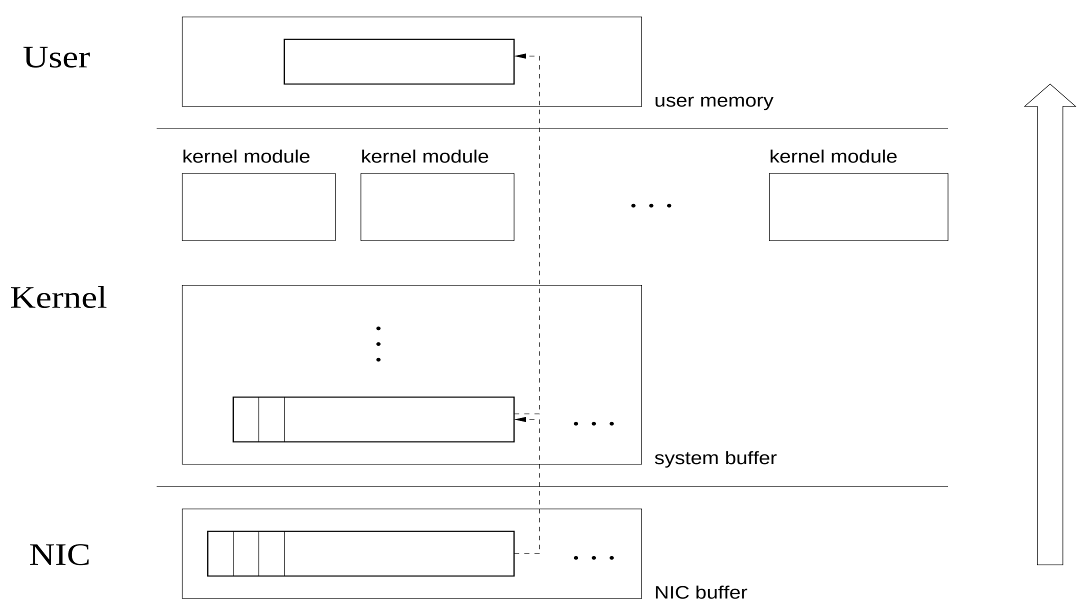
- decrement first element
- if equal to 0, then process associated event
- return

Copy reduction & horizontal design:

- keep copy operation to minimum
- use shared memory with pointers

  $\longrightarrow$ pointer passing
- use horizontal design to achieve parallelism

  $\longrightarrow$ multi-threading; sacrifice logical modularity



User space memory management.

⟶ data structure: e.g., trie, hashing for IP table

⟶ 200,000+ route entries

⟶ garbage collection

Keep number of system calls small.

⟶ system call is costly

⟶ stay in user space, if possible

Disk I/O.