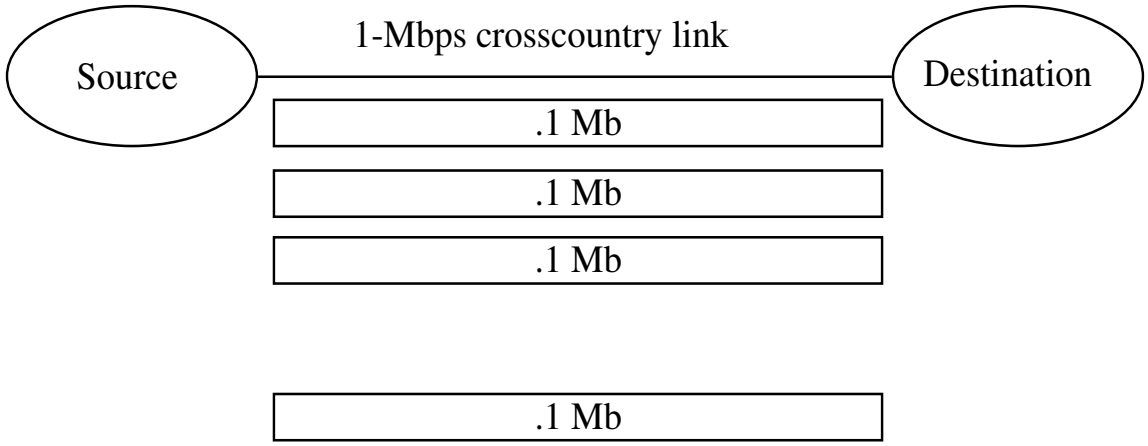
Copyright 1996. All rights reserved. Permission to reproduce this document for not for profit educational purposes is hereby granted. This document may not be reproduced for commercial purposes without the express written consent of the author.

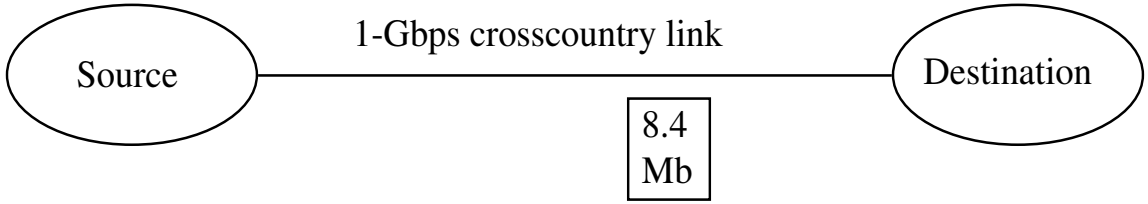
High-Speed Networks

Overview

- 1Mbps and 1Gbps cross-country links have the same latency
  - limited by the speed of light
- To transfer a 1MB file takes...
  - 100 RTTs on a 1Mbps link
  - doesn't fill a 1Gbps link (12.5MB delay x bandwidth)
- Said another way:
  - 1MB file is to 1Gbps network what 1KB packet is to 1Mbps network


Source
1-Mbps crosscountry link
Destination
.1 Mb
.1 Mb
.1 Mb
.1 Mb

(a)


Source
1-Gbps crosscountry link
Destination
8.4
Mb

(b)

Latency/Bandwidth Tradeoff

- **`Throughput = TransferSize / TransferTime`**
  - if it takes 10ms to transfer 1MB, then the effective throughput is 1MB/10ms = 100MBps = 800Mbps

- **`TransferTime = Latency + 1/Bandwidth` x `TransferSize`**
  - if network bandwidth is 1Gbps (it takes 1/1Gbps x 1MB = 0.8ms to transmit 1MB), an end-to-end transfer that requires 1 RTT of 100ms has a total transfer time of 100.8ms
  - effective throughput is 1MB/100.8ms = 79.4Mbps, not 1Gbps

- Notes
  - transferring a large amount of data helps improve the effective throughput; in the limit, an infinitely large transfer size causes the effective throughput to approach the network bandwidth
  - having to endure more than one RTT will hurt the effective throughput for any transfer of finite size, and will be most noticeable for small transfers

Implications

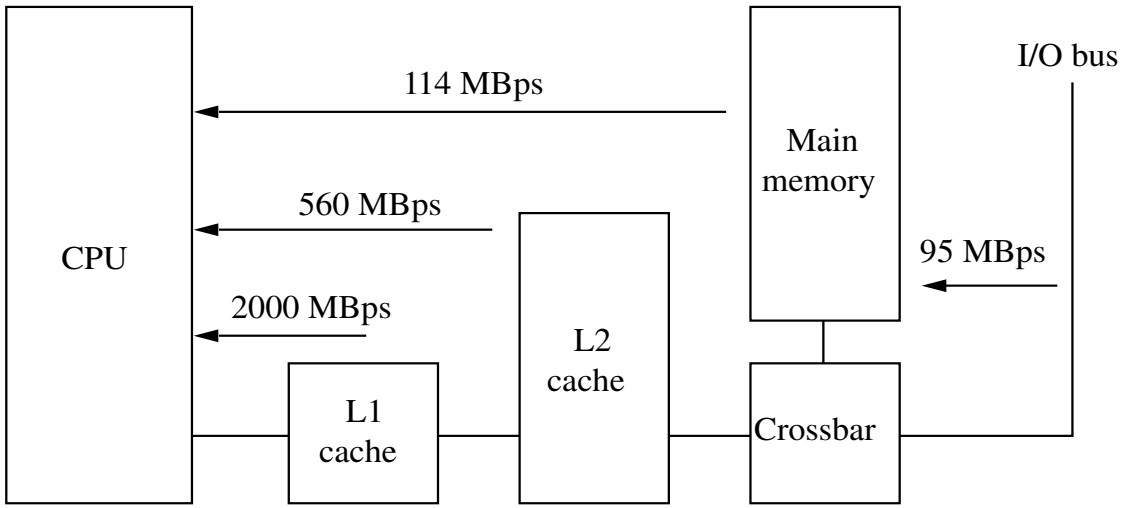
- Congestion control
  - feedback based mechanisms require an RTT to adjust
  - can send 10MB in one 100ms RTT on a 1Gbps network
  - that 10MB might congest a router and lead to massive losses
  - can lose half a delay x bandwidth's of data during slow start
  - reservations work for continuous streams (e.g., video), but require an extra RTT for bulk transfers

- Retransmissions
  - retransmitting a packet costs 1 RTT
  - dropping even one packet (cell) halves effective bandwidth
  - retransmission also implies buffering at the sender
  - possible solution: forward error correction (FEC)

- Trading bandwidth for latency
  - each RTT is precious
  - willing to “waste” bandwidth to save latency
  - example: prefetching

Host Memory Bottleneck

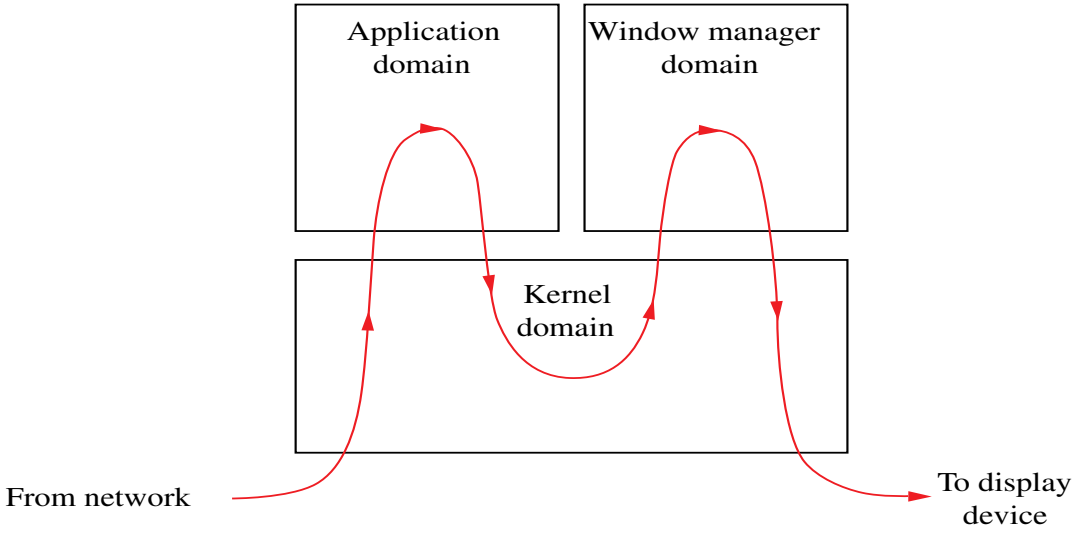
- Issue
  - turning host-to-host bandwidth into application-to-application bandwidth
  - have to deliver data across I/O and memory buses into cache and registers



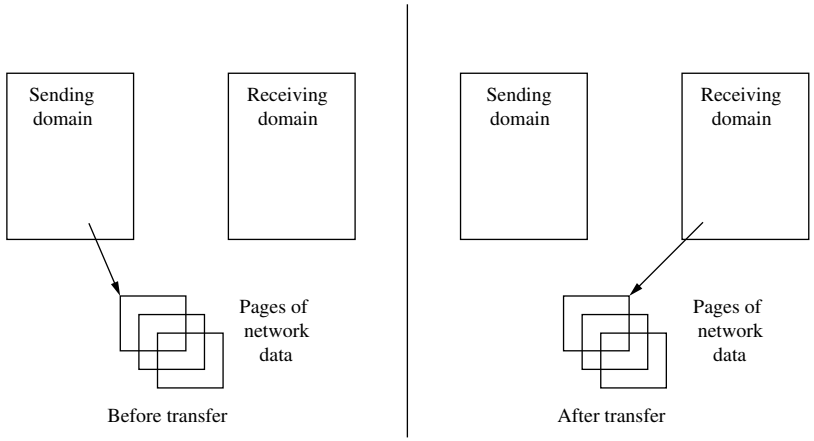
- Memory bandwidth
  - I/O bus must keep up with network speed (currently does for STS-12, assuming peak rate is achievable)
  - 114MBps (measured number) is only slightly faster than I/O bus; can't afford to go across memory bus twice
  - caches are of questionable value (rather small)
  - lots of reason to access buffers
    - user/kernel boundary
    - certain protocols (reassembly, checksumming)
    - network device and its driver
- Same latency/bandwidth problems as high-speed networks

Techniques for Avoiding Data Transfers

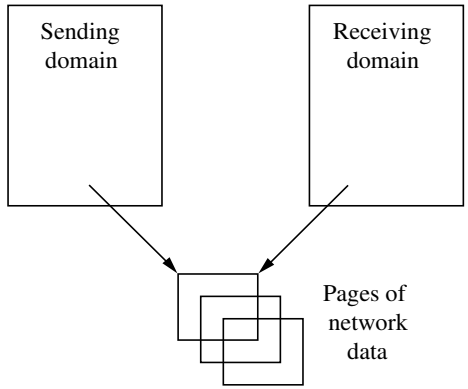
- Device/memory transfers
  - mechanism: DMA vs PIO
  - transfer size: cell vs packet
- Cross-domain transfers



– page remapping



– shared memory



– dynamic page sharing (fbufs)

- Data manipulations
  - compression, encryption, checksum, presentation formatting
  - involves memory loads and stores
  - integrated layer processing (ILP)

```
for (i=0; i<10000; i++)
    msgData[i]++;                /* LOAD, ADD, STORE */
for (i=0; i<10000; i++)
    msgData[i] = ~msgData[i];    /* LOAD, COMPLEMENT, STORE */

for (i=0; i<10000; i++)
    temp = msgData[i];           /* LOAD */
    temp++;                      /* ADD */
    temp = ~temp;                /* COMPLEMENT */
    msgData[i] = temp;           /* STORE */
```

- API design
  - write: application reuses buffer
  - read: application specifies buffer

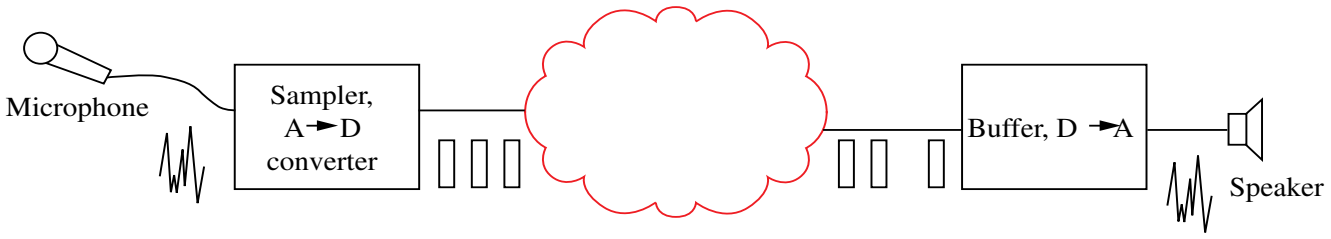
Integrated Services

- High-speed networks have enabled new applications
  - they also need “deliver on time” assurances from the network
- Applications that are sensitive to the timeliness of data are called *real-time* applications
  - voice
  - video
  - industrial control

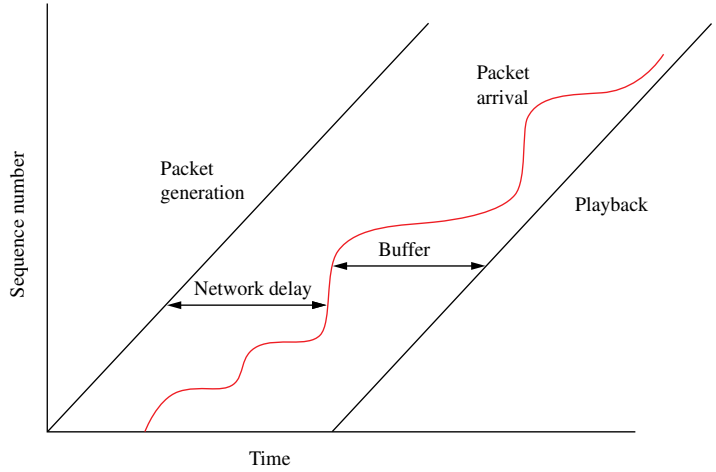
- Timeliness guarantees must come from *inside* the network
  - end-hosts cannot correct late packets like they can correct for lost packets
- Need more than best-effort
  - IETF is standardizing extensions to best-effort model

Service Model

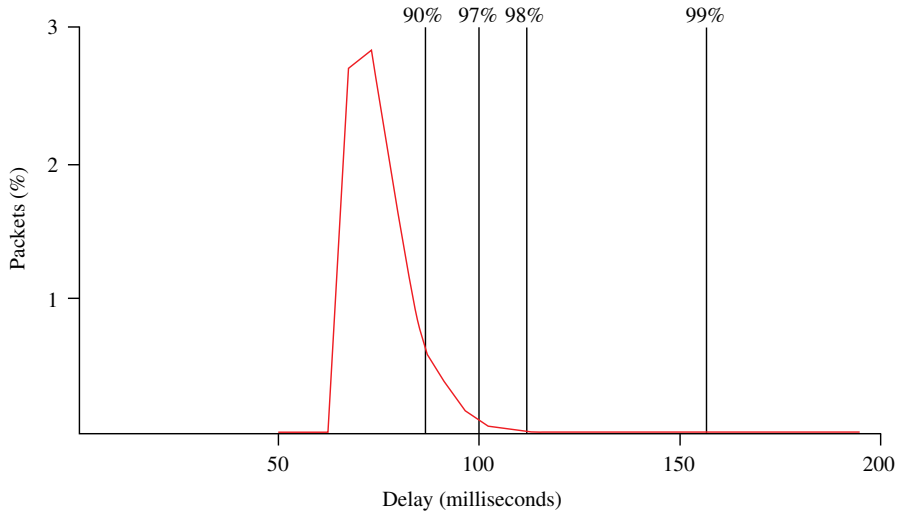
- Two types of applications
  - real-time
  - elastic
- Example real-time application (audio)
  - sample voice once every 125μs
  - each sample has a *playback time*
  - packets experience variable delay in network
  - add constant factor to playback time: *playback point*



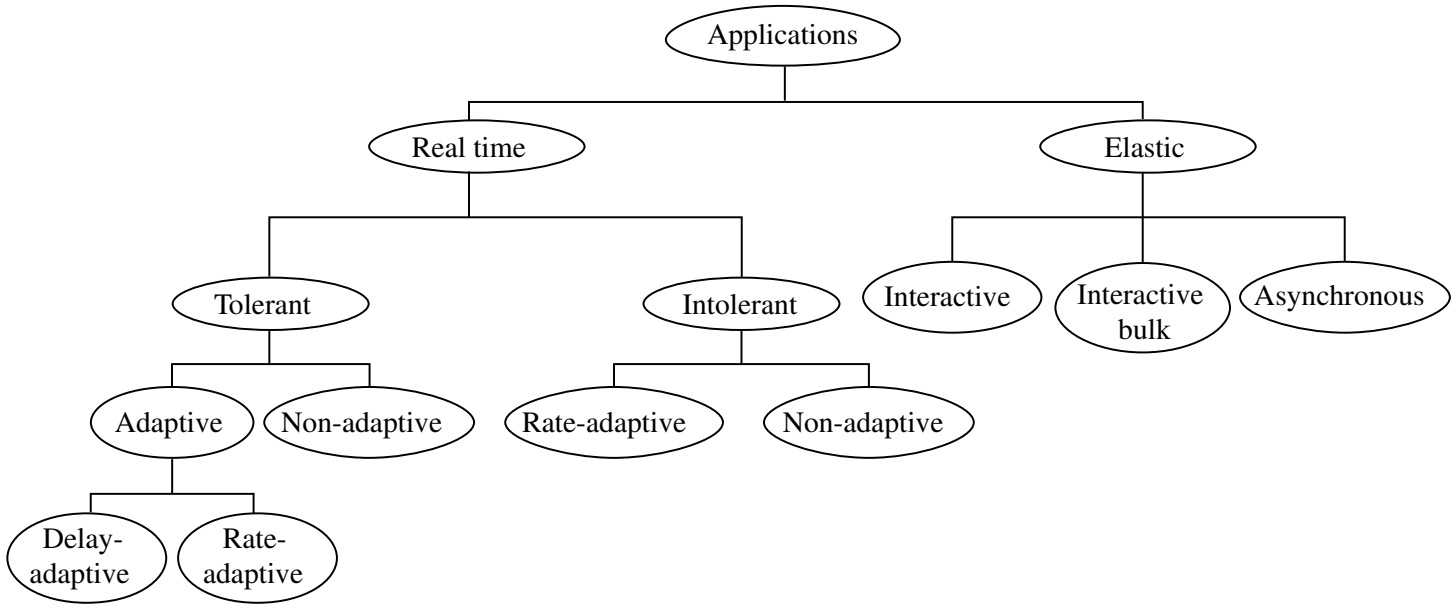
- Playback buffer



- Example distribution of delays



- Taxonomy of real-time applications



- Service classes
  - guaranteed
  - predictive
  - controlled-delay

Mechanisms

- Flowspecs
  - *Rspec*: describes service requested from network
    - controlled-delay: level of delay required
    - guaranteed/predictive: delay target
  - *Tspec*: describes flow's traffic characteristics
    - average bandwidth + burstiness: *token bucket* filter
    - token rate *r*
    - bucket depth *B*
    - must have a token to send a byte
    - must have *n* tokens to send *n* bytes
    - start with no tokens
    - accumulate tokens at rate of *r* per second
    - can accumulate no more than *B* tokens

- Admission Control
  - decide if a new flow can be supported
  - answer depends on service class
  - not the same as *policing*
- Reservation Protocol
  - called signalling in ATM
  - proposed Internet standard: RSVP
  - consistent with robustness of today's connectionless model
  - uses soft state (refresh periodically)
  - designed to support multicast
  - is receiver oriented
  - two messages: PATH and RESV
  - source transmits PATH messages sent every 30 seconds
  - destination responds with RESV message
  - merge requirements in case of multicast
  - can specify number of speakers

- Packet Classifying and Scheduling
  - associate each packet with the appropriate reservation so that it can be handled correctly, a process known as *classifying* packets
  - manage the packets in the queues so that they receive the service that has been requested, a process known as *packet scheduling*

RSVP versus ATM (Q.2931)

- RSVP
  - receiver generates reservation
  - soft state (refresh/timeout)
  - separate from route establishment
  - QoS can change dynamically
  - receiver heterogeneity
- ATM
  - sender generates connection request
  - hard state (explicit delete)
  - concurrent with route establishment
  - QoS is static for life of connection
  - uniform QoS to all receivers