Multimedia Networking

Instructor: Anirban Mahanti
Office: ICT 745
Email: mahanti@cpsc.ucalgary.ca
Class Location: ICT 121
Lectures: MWF 12:00 – 12:50 hours
Notes derived from "*Computer Networking: A Top Down Approach Featuring the Internet*", 2005, 3rd edition, Jim Kurose, Keith Ross, Addison-Wesley.

Slides are adapted from the companion web site of the book, as modified by Anirban Mahanti (and Carey Williamson).

Goals

Principles
- Classify multimedia applications
- Identify the network services the apps need
- Making the best of best effort service
- Mechanisms for providing QoS

Protocols and Architectures
- Specific protocols for best-effort
- Architectures for QoS

Why Study Multimedia Networking?

- Exciting, industry relevant research topic
- Multimedia is everywhere
- Tons of open problems

Outline

- Multimedia Networking Applications
  - Stored, live, interactive
  - Multimedia over "Best Effort" Internet
  - Evolving the Internet to support multimedia applications
- Streaming stored audio and video
- Scalable Streaming Techniques (Hot Topic)
- Content Distribution Networks (Hot Topic)
- Beyond Best Effort

MM Networking Applications

Classes of MM applications:

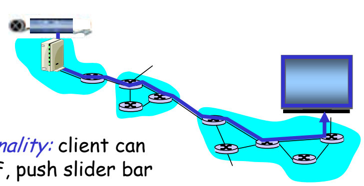
1) Streaming stored audio and video
2) Streaming live audio and video
3) Real-time interactive audio and video

Jitter is the variability of packet delays within the same packet stream

Fundamental characteristics:

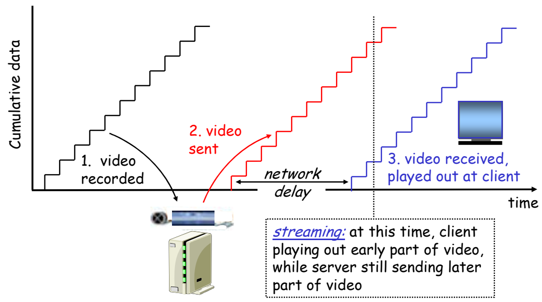
- Typically **delay sensitive**
  - end-to-end delay
  - delay jitter
- But **loss tolerant**: infrequent losses cause minor glitches
- *Antithesis* of data, which are loss intolerant but delay tolerant.

Streaming Stored Multimedia (1/2)



- *VCR-like functionality:* client can pause, rewind, FF, push slider bar
  - 10 sec initial delay OK
  - 1-2 sec until command effect OK
  - need a separate control protocol?
- timing constraint for still-to-be transmitted data: in time for playout

Streaming Stored Multimedia (2/2)



Streaming Live Multimedia

Examples:
- Internet radio talk show
- Live sporting event

Streaming
- playback buffer
- playback can lag tens of seconds after transmission
- still have timing constraint

Interactivity
- fast forward impossible
- rewind, pause possible!

Interactive, Real-Time Multimedia

- applications: IP telephony, video conference, distributed interactive worlds
- end-end delay requirements:
  - audio: < 150 msec good, < 400 msec OK
    - includes application-layer (packetization) and network delays
    - higher delays noticeable, impair interactivity
- session initialization
  - how does callee advertise its IP address, port number, encoding algorithms?

Multimedia Over "Best Effort" Internet

- TCP/UDP/IP: *no* guarantees on delay, loss

? ? ? ? ? ?
But you said multimedia apps requires ?
QoS and level of performance to be
? effective! ? ?

Today's multimedia applications implement functionality at the app. layer to mitigate (as best possible) effects of delay, loss

How to provide better support for Multimedia? (1/4)

Integrated services philosophy:

- architecture for providing QOS guarantees in IP networks for individual flows
- Fundamental changes in Internet so that apps can reserve end-to-end bandwidth
- Components of this architecture are
  - Admission control
  - Reservation protocol
  - Routing protocol
  - Classifier and route selection
  - Packet scheduler

How to provide better support for Multimedia? (2/4)

Concerns with Intserv:

- Scalability: signaling, maintaining per-flow router state difficult with large number of flows
- Flexible Service Models: Intserv has only two classes. Desire "qualitative" service classes
  - E.g., Courier, xPress, and normal mail
  - E.g., First, business, and cattle class ☺

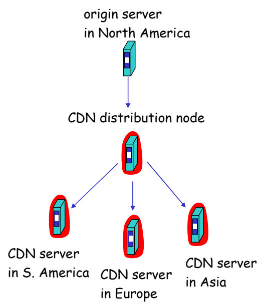
Diffserv approach:

- simple functions in network core, relatively complex functions at edge routers (or hosts)
- Don't define define service classes, provide functional components to build service classes

How to provide better support for Multimedia? (3/4)

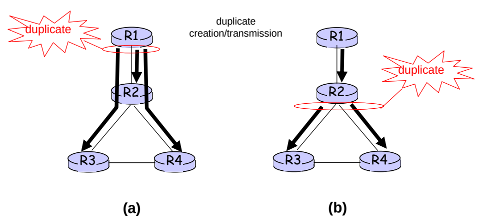
Content Distribution Networks (CDNs)

- Challenging to stream large files (e.g., video) from single origin server in real time
- Solution: replicate content at hundreds of servers throughout Internet
  - content downloaded to CDN servers ahead of time
  - placing content "close" to user avoids impairments (loss, delay) of sending content over long paths
  - CDN server typically in edge/access network



How to provide better support for Multimedia? (4/4)

Multicast/Broadcast

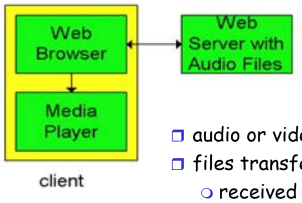


Source-duplication versus in-network duplication.
(a) source duplication, (b) in-network duplication

Outline

- Multimedia Networking Applications
- Streaming stored audio and video
  - Streaming Architectures
  - Real Time Streaming Protocol
  - Packet Loss Recovery
- Streaming stored audio and video
- Scalable Streaming Techniques (Hot Topic)
- Content Distribution Networks (Hot Topic)
- Beyond Best Effort

Internet multimedia: simplest approach

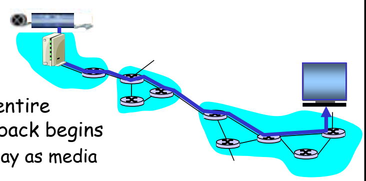


- audio or video stored in file
- files transferred as HTTP object
  - received in entirety at client
  - then passed to player

audio, video not streamed:

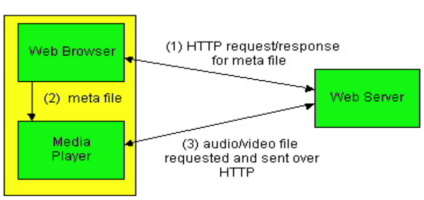
- no, "pipelining," long delays until playout!

Streaming vs. Download of Stored Multimedia Content



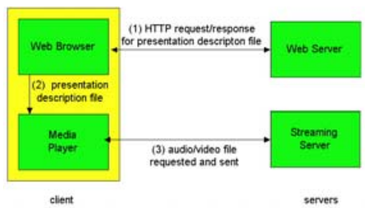
- Download: Receive entire content before playback begins
  - High "start-up" delay as media file can be large
  - ~ 4GB for a 2 hour MPEG II movie
- Streaming: Play the media file while it is being received
  - Reasonable "start-up" delays
  - Reception Rate >= playback rate. Why?

Progressive Download



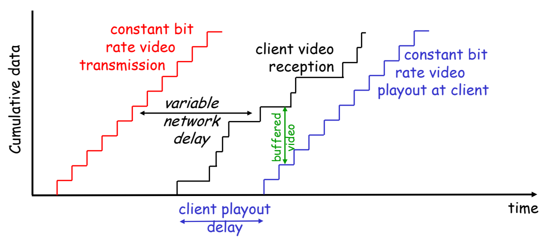
- browser GETs **metafile**
- browser launches player, passing metafile
- player contacts server
- server downloads audio/video to player

Streaming from a Streaming Server



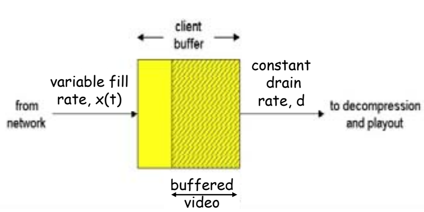
- This architecture allows for non-HTTP protocol between server and media player
- Can also use UDP instead of TCP.
- Example: Browse the Helix product family
  http://www.realnetworks.com/products/media_delivery.html

Streaming Multimedia: Client Buffering



- Client-side buffering, playout delay compensate for network-added delay, delay jitter

Streaming Multimedia: Client Buffering



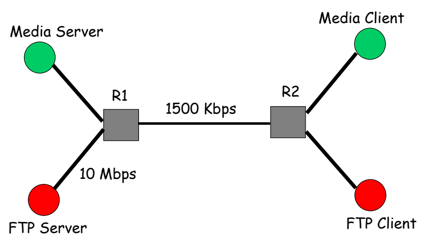
- Client-side buffering, playout delay compensate for network-added delay, delay jitter

Streaming Multimedia: UDP or TCP?

UDP
- server sends at rate appropriate for client (oblivious to network congestion !)
  - often send rate = encoding rate = constant rate
  - then, fill rate = constant rate - packet loss
- short playout delay (2-5 seconds) to compensate for network delay jitter
- error recover: time permitting

TCP
- send at maximum possible rate under TCP
- fill rate fluctuates due to TCP congestion control
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls

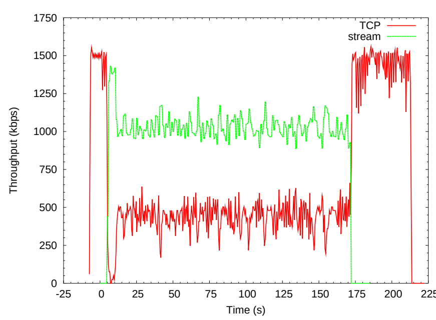
Fairness of RealVideo Streams (1/2)



- R1-R2 is the bottleneck link
- Media Server is DNA Helix Server from RealNetworks
- Streaming uses UDP at the transport layer; requesting media encoded at 1 Mbps
- What fraction of the bottleneck is available to FTP?

Talk to Sean Boyden if you want to know more ☺

Fairness of RealVideo Streams (2/2)



Outline

- Multimedia Networking Applications
- Streaming stored audio and video
  - Streaming Architectures
  - Real Time Streaming Protocol
  - Packet Loss Recovery
- Streaming stored audio and video
- Scalable Streaming Techniques (Hot Topic)
- Content Distribution Networks (Hot Topic)
- Beyond Best Effort

Real-Time Streaming Protocol (RTSP)

HTTP

- Does not target multimedia content
- No commands for fast forward, etc.

RTSP: RFC 2326

- Client-server application layer protocol.
- For user to control display: rewind, fast forward, pause, resume, repositioning, etc…

What it doesn't do:

- does not define how audio/video is encapsulated for streaming over network
- does not restrict how streamed media is transported; it can be transported over UDP or TCP
- does not specify how the media player buffers audio/video

RTSP Example

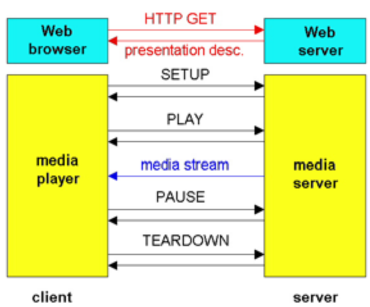
Scenario:

- metafile communicated to web browser
- browser launches player
- player sets up an RTSP control connection, data connection to streaming server

Metafile Example

```
<title>Twister</title>
<session>
      <group language=en lipsync>
            <switch>
              <track type=audio
                   e="PCMU/8000/1"
                   src = "rtsp://audio.example.com/twister/audio.en/lofi">
              <track type=audio
                   e="DVI4/16000/2" pt="90 DVI4/8000/1"
                   src="rtsp://audio.example.com/twister/audio.en/hifi">
            </switch>
          <track type="video/jpeg"
                   src="rtsp://video.example.com/twister/video">
       </group>
</session>
```


RTSP Operation



RTSP Exchange Example

C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
Transport: rtp/udp; compression; port=3056; mode=PLAY

S: RTSP/1.0 200 1 OK
Session 4231

C: PLAY rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=0-

C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=37

C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231

S: 200 3 OK

Outline

- Multimedia Networking Applications
- Streaming stored audio and video
  - Streaming Architectures
  - Real Time Streaming Protocol
  - Packet Loss Recovery
- Streaming stored audio and video
- Scalable Streaming Techniques (Hot Topic)
- Content Distribution Networks (Hot Topic)
- Beyond Best Effort

Packet Loss

- network loss: IP datagram lost due to network congestion (router buffer overflow)
- delay loss: IP datagram arrives too late for playout at receiver
  - delays: processing, queueing in network; end-system (sender, receiver) delays
  - Tolerable delay depends on the application
- How can packet loss be handled?
  - We will discuss this next …

Receiver-based Packet Loss Recovery

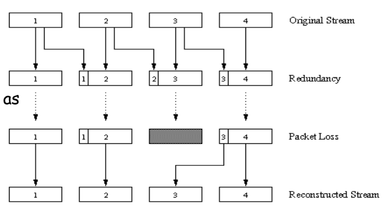
- Generate replacement packet
  - Packet repetition
  - Interpolation
  - Other sophisticated schemes
- Works when audio/video stream exhibits short-term self-similarity
- Works for relatively low loss rates (e.g., < 5%)
- Typically, breaks down on "bursty" losses

Forward Error Correction (FEC)

- for every group of n packets generate k redundant packets
- send out n+k packets, increasing the bandwidth by factor k/n.
- can reconstruct the original n packets provided at most k packets are lost from the group
- Works well at high loss rate (for a proper choice of k)
- Handles “bursty” packet losses
- Cost: increase in transmission cost (bandwidth)

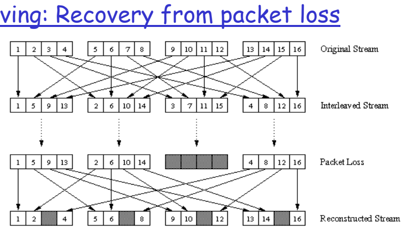
Another FEC Example

- “piggyback lower quality stream”
- Example: send lower resolution audio stream as the redundant information
-



- Whenever there is non-consecutive loss, the receiver can conceal the loss.
- Can also append (n-1)st and (n-2)nd low-bit rate chunk

Interleaving: Recovery from packet loss



Interleaving
- Re-sequence packets before transmission
- Better handling of “burst” losses
- Results in increased playout delay

Summary: Internet Multimedia: bag of tricks

- use UDP to avoid TCP congestion control (delays) for time-sensitive traffic
- client-side adaptive playout delay: to compensate for delay
- server side matches stream bandwidth to available client-to-server path bandwidth
  - chose among pre-encoded stream rates
  - dynamic server encoding rate
- error recovery (on top of UDP)
  - FEC, interleaving
  - retransmissions, time permitting
  - conceal errors: repeat nearby data

Outline

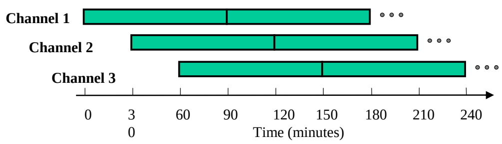
- Multimedia Networking Applications
- Streaming stored audio and video
- Scalable Streaming Techniques
- Content Distribution Networks
- Beyond Best Effort

Streaming Popular Content

- Consider a popular media file
  - Playback rate: 1 Mbps
  - Duration: 90 minutes
  - Request rate: once every minute
- How can a video server handle such high loads?
  - Approach 1: Start a new "stream" for each request
  - Allocate server and disk I/O bandwidth for each request
  - Bandwidth required at server= 1 Mbps x 90

Streaming Popular Content using Batching

- Approach 2: Leverage the multipoint delivery capability of modern networks
- Playback rate = 1 Mbps, duration = 90 minutes
- Group requests in non-overlapping intervals of 30 minutes:
  - Max. start-up delay = 30 minutes
  - Bandwidth required = 3 channels = 3 Mbps

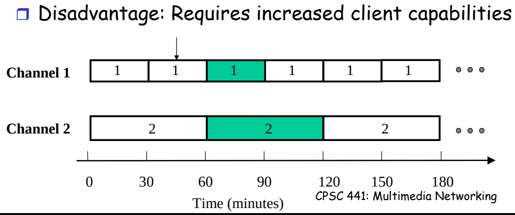


Batching Issues

- Bandwidth increases linearly with decrease in start-up delays
- Can we reduce or eliminate "start-up" delays?
  - Periodic Broadcast Protocols
  - Stream Merging Protocols

Periodic Broadcast Example

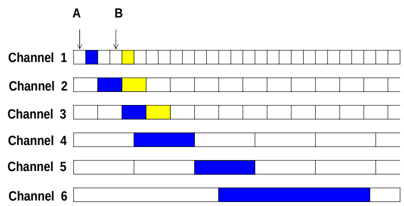
- Partition the media file into 2 segments with relative sizes {1, 2}. For a 90 min. movie:
  - Segment 1 = 30 minutes, Segment 2 = 60 minutes
- Advantage:
  - Max. start-up delay = 30 minutes
  - Bandwidth required = 2 channels = 2 Mbps
- Disadvantage: Requires increased client capabilities



Skyscraper Broadcasts (SB)

[Hua & Sheu 1997]

- Divide the file into *K* segments of increasing size
  - Segment size progression: 1, 2, 2, 5, 5, 12, 12, 25, …
- Multicast each segment on a separate channel at the playback rate
- Aggregate rate to clients: *2 x playback rate*



Comparing Batching and SB

Server Bandwidth	Start-up Delay	
	Batching	SB
1 Mbps	90 minutes	90 minutes
2 Mbps	45 minutes	30 minutes
6 Mbps	15 minutes	3 minutes
10 Mbps	9 minutes	30 seconds

- Playback rate = 1 Mbps, duration = 90 minutes
- Limitations of Skyscraper:
  - Ad hoc segment size progress
  - Does not work for low client data rates

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al.* 2001, 2003, 2004]

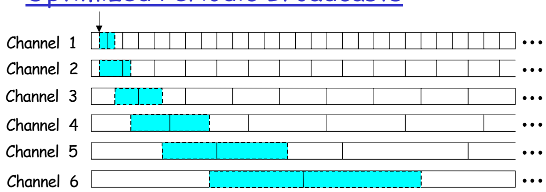
- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend for packet loss recovery
- extend for "bursty" packet loss
- extend for client heterogeneity

Reliable Periodic Broadcasts (RPB)

[Mahanti *et al.* 2001, 2003, 2004]

- Optimized PB protocols (no packet loss recovery)
  - client fully downloads each segment before playing
  - required server bandwidth near minimal
  - Segment size progression is *not* ad hoc
  - Works for client data rates < 2 x playback rate
- extend  loss recovery
- extend y" packet loss
- extend heterogeneity

Optimized Periodic Broadcasts



- r = segment streaming rate = 1
- s = maximum # streams client listens to concurrently = 2
- b = client data rate = s x r = 2
- length of first s segments: $\frac{1}{r} l_k = \frac{1}{r} l_1 + \sum_{j=1}^{k-1} l_j$
- length of segment k > s: $\frac{1}{r} l_k = \sum_{j=k-s}^{k-1} l_j$

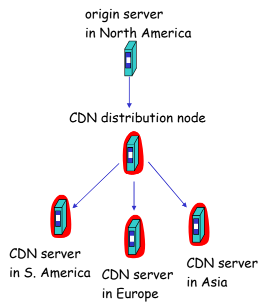
Outline

- Multimedia Networking Applications
- Streaming stored audio and video
- Scalable Streaming Techniques
- Content Distribution Networks
- Beyond Best Effort

Content distribution networks (CDNs)

Content replication

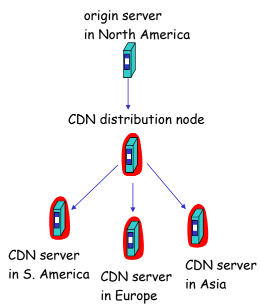
- Challenging to stream large files (e.g., video) from single origin server in real time
- Solution: replicate content at hundreds of servers throughout Internet
  - content downloaded to CDN servers ahead of time
  - placing content "close" to user avoids impairments (loss, delay) of sending content over long paths
  - CDN server typically in edge/access network



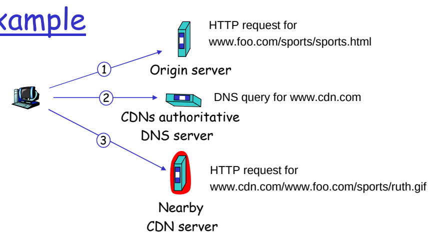
Content distribution networks (CDNs)

Content replication

- CDN (e.g., Akamai) customer is the content provider (e.g., CNN)
- CDN replicates customers' content in CDN servers. When provider updates content, CDN updates servers



CDN example



origin server (www.foo.com)

- distributes HTML
- replaces:
  http://www.foo.com/sports.ruth.gif
  with
  http://www.cdn.com/www.foo.com/sports/ruth.gif

CDN company (cdn.com)

- distributes gif files
- uses its authoritative DNS server to route redirect requests

More about CDNs

routing requests

- CDN creates a "map", indicating distances from leaf ISPs and CDN nodes
- when query arrives at authoritative DNS server:
  - server determines ISP from which query originates
  - uses "map" to determine best CDN server
- CDN nodes create application-layer overlay network

Outline

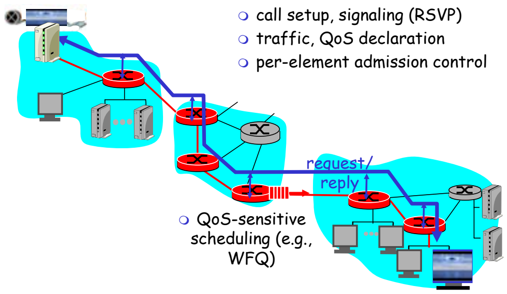
- Multimedia Networking Applications
- Streaming stored audio and video
- Scalable Streaming Techniques
- Content Distribution Networks
- Beyond Best Effort

Integrated Services (Intserv) Architecture

- architecture for providing QOS guarantees in IP networks for individual flows
- flow: a distinguishable stream of distinct IP datagrams
  - Unidirectional
  - Multiple recipient
- Components of this architecture:
  - Admission control
  - Reservation protocol
  - Routing protocol
  - Classifier and route selection
  - Packet scheduler

Intserv: QoS guarantee scenario

- Resource reservation
  - call setup, signaling (RSVP)
  - traffic, QoS declaration
  - per-element admission control



- QoS-sensitive scheduling (e.g., WFQ)

Call Admission

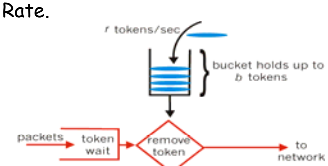
Arriving session must :

- declare its QOS requirement
  - R-spec: defines the QOS being requested
- characterize traffic it will send into network
  - T-spec: defines traffic characteristics
- signaling protocol: needed to carry R-spec and T-spec to routers (where reservation is required)
  - RSVP

Need Scheduling and Policing Policies to provide QoS

Policing: Token Bucket

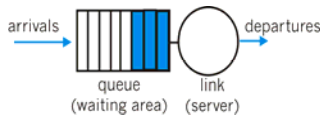
Token Bucket: limit input to specified Burst Size and Average Rate.



- bucket can hold b tokens
- tokens generated at rate *r token/sec* unless bucket full
- *over interval of length t: number of packets admitted less than or equal to (r t + b).*

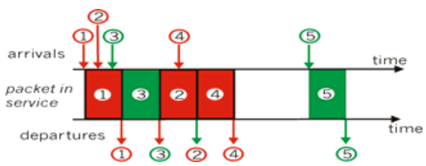
Link Scheduling

- scheduling: choose next packet to send on link
- FIFO (first in first out) scheduling: send in order of arrival to queue
  - real-world example?
  - discard policy: if packet arrives to full queue: who to discard?
    - Tail drop: drop arriving packet
    - priority: drop/remove on priority basis
    - random: drop/remove randomly



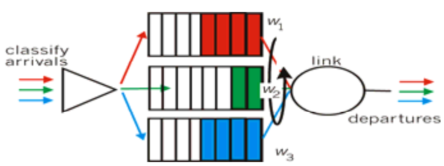
Round Robin

- multiple classes
- cyclically scan class queues, serving one from each class (if available)
- real world example?



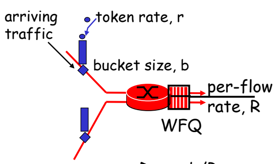
Weighted Fair Queuing

- generalized Round Robin
- each class gets weighted amount of service in each cycle
- real-world example?



Intserv QoS: Service models [rfc2211, rfc 2212]

Guaranteed service:
- Assured data rate
- A specified upper bound on queuing delay



$D_{max} = b/R$

Controlled load service:
- "a quality of service closely approximating the QoS that same flow would receive from an unloaded network element."
- Similar to behavior best effort service in an unloaded network

Differentiated Services

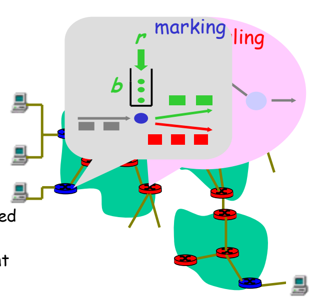
Concerns with Intserv:
- Scalability: signaling, maintaining per-flow router state difficult with large number of flows
- Flexible Service Models: Intserv has only two classes. Desire "qualitative" service classes
  - E.g., Courier, xPress, and normal mail
  - E.g., First, business, and cattle class ☺

Diffserv approach:
- simple functions in network core, relatively complex functions at edge routers (or hosts)
- Don't define define service classes, provide functional components to build service classes

Diffserv Architecture

Edge router:
- per-flow traffic management
- Set the DS field; value determines type of service

Core router:
- buffering and scheduling based on marking at edge
- per-class traffic management



Traffic Classification/Conditioning

- How can packet marks be carried in IPv4 datagrams?
- Sender may agree to conform to a "traffic profile", thus a leaky bucket policer may be used at the network edge to enforce
  - Peak rate
  - Average rate
  - Burst size
- What happens when traffic profile is violated?
  - Employ traffic shaping?

Forwarding (PHB)

- PHB result in a different observable (measurable) forwarding performance behavior
- PHB does not specify what mechanisms to use to ensure required PHB performance behavior
- Examples:
  - Class A gets x% of outgoing link bandwidth over time intervals of a specified length
  - Class A packets leave first before packets from class B

PHB's Defined in Diffserv

- Expedited Forwarding: pkt departure rate of a class equals or exceeds specified rate
  - logical link with a minimum guaranteed rate
- Assured Forwarding: 4 classes of traffic
  - each guaranteed minimum amount of bandwidth
  - each with three drop preference partitions

Deployment Issues

- Single administrative domain
- Incremental deployment
- Traffic policing/shaping complexity
- Charging models

Signaling in the Internet

connectionless (stateless) forwarding by IP routers + best effort service = no network signaling protocols in initial IP design

- New requirement: reserve resources along end-to-end path (end system, routers) for QoS for multimedia applications
- RSVP: Resource Reservation Protocol [RFC 2205]
  - " … allow users to communicate requirements to network in robust and efficient way." i.e., signaling !
- earlier Internet Signaling protocol: ST-II [RFC 1819]

RSVP Design Goals

1. accommodate heterogeneous receivers (different bandwidth along paths)
2. accommodate different applications with different resource requirements
3. make multicast a first class service, with adaptation to multicast group membership
4. leverage existing multicast/unicast routing, with adaptation to changes in underlying unicast, multicast routes
5. control protocol overhead to grow (at worst) linear in # receivers
6. modular design for heterogeneous underlying technologies

RSVP: does not…

- specify how resources are to be reserved
  - rather: a mechanism for communicating needs
- determine routes packets will take
  - that's the job of routing protocols
  - signaling decoupled from routing
- interact with forwarding of packets
  - separation of control (signaling) and data (forwarding) planes

Multimedia Networking: Summary

- multimedia applications and requirements
- making the best of today's best effort service
- scheduling and policing mechanisms
- next generation Internet: Intserv, RSVP, Diffserv