Chapter 6: Multimedia Networking

Our goals:
- principles: network, application-level support for multimedia
  - different forms of network multimedia, requirements
  - making the best of best effort service
  - mechanisms for providing QoS
- specific streaming protocols
- architectures for QoS

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

Multimedia, Quality of Service: What is it?

QoS
network provides application with level of performance needed for application to function.
**Multimedia Performance Requirements**

**Requirement:** deliver data in “timely” manner

- interactive multimedia: short end-end delay
  - e.g., IP telephony, teleconf., virtual worlds, DIS
  - excessive delay impairs human interaction
- streaming (non-interactive) multimedia:
  - data must arrive in time for “smooth” playout
  - late arriving data introduces gaps in rendered audio/video

- reliability: 100% reliability not always required

---

**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

**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

**Jitter** is the variability of packet delays within the same packet stream
Streaming Stored Multimedia

Streaming:
- media stored at source
- transmitted to client
- streaming: client playout begins before all data has arrived
- timing constraint for still-to-be transmitted data: in time for playout

Streaming Stored Multimedia: What is it?

Cumulative data

1. video recorded
2. video sent
3. video received, played out at client

Streaming: at this time, client playing out early part of video, while server still sending later part of video
Streaming Multimedia - Interactivity

Types of interactivity:
- **none**: like broadcast radio, TV
  - initial startup delays of < 10 secs OK
- **VCR-functionality**: client can pause, rewind, FF
  - 1-2 sec until command effect OK
- **timing constraint** for still-to-be transmitted data: in time for playout

Streaming Live Multimedia

**Examples:**
- Internet radio talk show
- Live sporting event (e.g., soccer game)

**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-level (packetization) and network delays
    - Higher delays noticeable, impair interactivity
- **Session initialization**
  - How does callee advertise its IP address, port number, encoding algorithms?

Multimedia Over Today’s Internet

**TCP/UDP/IP**: “best-effort service”

- **No** guarantees on delay, loss

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

Today’s Internet multimedia applications use application-level techniques to mitigate (as best possible) effects of delay, loss
How should the Internet evolve to better support multimedia?

**Integrated services philosophy:**
- Fundamental changes in Internet so that apps can reserve end-to-end bandwidth
- Requires new, complex software in hosts & routers

**Laissez-faire**
- no major changes
- more bandwidth when needed
- content distribution, application-layer mechanisms
  - application layer

**Differentiated services philosophy:**
- Fewer changes to Internet infrastructure, yet provide 1st and 2nd class service.

What's your opinion?

---

**Streaming Stored Multimedia**

Application-level streaming techniques for making the best out of best effort service:
- client side buffering
- use of UDP versus TCP
- multiple encodings of multimedia

**MediaPlayer**
- jitter removal
- decompression
- error concealment
- graphical user interface w/ controls for interactivity
**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!

**Internet multimedia: streaming approach**

- browser GETs metafile
- browser launches player, passing metafile
- player contacts server
- server streams 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.

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 recovery: time permitting

**TCP**
- send at maximum possible rate under TCP
- congestion loss: fill rate fluctuates
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls
Streaming Multimedia: client rate(s)

**Q:** how to handle different client receive rate capabilities?
- 28.8 Kbps dialup
- 100Mbps Ethernet

**A:** server stores, transmits multiple copies of video, encoded at different rates

Real-time interactive applications

- PC-2-PC phone  
  - instant messaging services are providing this
- PC-2-phone  
  - Dialpad  
  - Net2phone
- videoconference with Webcams

Let's look at a PC-2-PC Internet phone example in detail.
**Interactive Multimedia: Internet Phone**

- Introduce Internet Phone by way of an example
  - speaker's audio: alternating talk spurts, silent periods.
    - 64 kbps during talk spurt
  - pkts generated only during talk spurts
    - 20 msec chunks at 8 Kbytes/sec: 160 bytes data
  - application-layer header added to each chunk
  - chunk+header encapsulated into UDP segment
  - application sends UDP segment into socket every 20 msec during talk spurt

**Internet Phone: Packet Loss and Delay**

- 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
  - typical maximum tolerable delay: 400 ms
- loss tolerance: depending on voice encoding, losses concealed, packet loss rates between 1% and 10% can be tolerated.
**Delay Jitter**

- Consider the end-to-end delays of two consecutive packets: difference can be more or less than 20 msec

---

**Internet Phone: Fixed Playout Delay**

- Receiver attempts to playout each chunk exactly $q$ msecs after chunk was generated.
  - chunk has time stamp $t$: play out chunk at $t+q$.
  - chunk arrives after $t+q$: data arrives too late for playout, data “lost”

- Tradeoff for $q$:
  - large $q$: less packet loss
  - small $q$: better interactive experience
**Fixed Playout Delay**

- Sender generates packets every 20 msec during talk spurt.
- First packet received at time $r$
- First playout schedule: begins at $p$
- Second playout schedule: begins at $p'$

**Recovery From Packet Loss**

- **loss**: packet never arrives or arrives too late
- **real-time constraints**: little (no) time for retransmissions!
  - What to do?
- **Forward Error Correction (FEC)**: add error correction bits (recall 2-dimensional parity)
  - add redundant chunk made up of exclusive OR of $n$ chunks
  - redundancy (overhead) is $1/n$
  - can reconstruct if at most one lost chunk
- **Interleaving**: spread loss evenly over received data to minimize impact of loss
FEC - Piggybacking Lower Quality Stream

**FEC Scheme:**
- "piggyback" lower quality stream
- 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

**Interleaving Scheme**
- no redundancy needed
- chunks are broken up into smaller units
  - for example, four 5 msec units per chunk
- packet contains small units from different chunks
- if packet is lost, still have most of every chunk
- has no redundancy overhead
- but adds to 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

Improving QoS in IP Networks

Thus far: “making the best of best effort”
Future: next generation Internet with QoS guarantees
  - RSVP: signaling for resource reservations
  - Differentiated Services: differential guarantees
  - Integrated Services: firm guarantees
- simple model for sharing and congestion studies:
Principles for QOS Guarantees

- Example: 1Mbps IP phone, FTP share 1.5 Mbps link.
  - bursts of FTP can congest router, cause audio loss
  - want to give priority to audio over FTP

Principle 1
packet marking needed for router to distinguish between different classes; and new router policy to treat packets accordingly

Principles for QOS Guarantees (more)

- what if applications misbehave (audio sends higher than declared rate)
  - policing: force source adherence to bandwidth allocations
- marking and policing at network edge:
  - similar to ATM UNI (User Network Interface)

Principle 2
provide protection (isolation) for one class from others
Principles for QOS Guarantees (more)

- Allocating fixed (non-sharable) bandwidth to flow: inefficient use of bandwidth if flows doesn’t use its allocation

While providing isolation, it is desirable to use resources as efficiently as possible

Principle 3

Basic fact of life: can not support traffic demands beyond link capacity

Call Admission: flow declares its needs, network may block call (e.g., busy signal) if it cannot meet needs
Summary of QoS Principles

Let's next look at mechanisms for achieving this ....

Scheduling And Policing Mechanisms

- **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

![Diagram of scheduling and policing mechanisms with a queue and link representing arrivals and departures](image)
**Scheduling Policies: more**

**Priority scheduling:** transmit highest priority queued packet
- multiple classes, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc..
  - Real world example?

**Scheduling Policies: still more**

round robin scheduling:
- multiple classes
- cyclically scan class queues, serving one from each class (if available)
- real world example?
Scheduling Policies: still more

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

Policing Mechanisms

**Goal:** limit traffic to not exceed declared parameters

Three common-used criteria:
- *(Long term) Average Rate:* how many pkts can be sent per unit time (in the long run)
  - crucial question: what is the interval length: 100 packets per sec or 6000 packets per min have same average!
- *Peak Rate:* e.g., 6000 pkts per min. (ppm) avg.; 1500 ppm peak rate
- *(Max.) Burst Size:* max. number of pkts sent consecutively (with no intervening idle)
Policing Mechanisms

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

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

Policing Mechanisms (more)

- token bucket, WFQ combine to provide guaranteed upper bound on delay, i.e., *QoS guarantee!*

\[ D_{\text{max}} = \frac{b}{R} \]
IETF Integrated Services

- architecture for providing QoS guarantees in IP networks for individual application sessions
- resource reservation: routers maintain state info (a la VC) of allocated resources, QoS req's
- admit/deny new call setup requests:

**Question:** can newly arriving flow be admitted with performance guarantees while not violated QoS guarantees made to already admitted flows?

**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

Intserv QoS: Service models [rfc2211, rfc 2212]

Guaranteed service:
- worst case traffic arrival: leaky-bucket-policed source
- simple (mathematically provable) bound on delay [Parekh 1992, Cruz 1988]

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

WFQ

\[ D_{max} = \frac{b}{R} \]
IETF 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. Also want “qualitative” service classes
  - “behaves like a wire”
  - relative service distinction: Platinum, Gold, Silver

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

Diffserv Architecture

**Edge router:**
- per-flow traffic management
- marks packets as *in-profile* and *out-profile*

**Core router:**
- per class traffic management
- buffering and scheduling based on marking at edge
- preference given to *in-profile* packets
- Assured Forwarding
**Edge-router Packet Marking**

- **profile**: pre-negotiated rate \( A \), bucket size \( B \)
- **packet marking at edge based on per-flow profile**

Possible usage of marking:
- **class-based marking**: packets of different classes marked differently
- **intra-class marking**: conforming portion of flow marked differently than non-conforming one

**Classification and Conditioning**

- Packet is marked in the Type of Service (TOS) in IPv4, and Traffic Class in IPv6
- 6 bits used for Differentiated Service Code Point (DSCP) and determine PHB that the packet will receive
- 2 bits are currently unused
**Classification and Conditioning**

may be desirable to limit traffic injection rate of some class:

- user declares traffic profile (eg, rate, burst size)
- traffic metered, shaped if non-conforming

**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
Forwarding (PHB)

PHBs being developed:
- **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

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