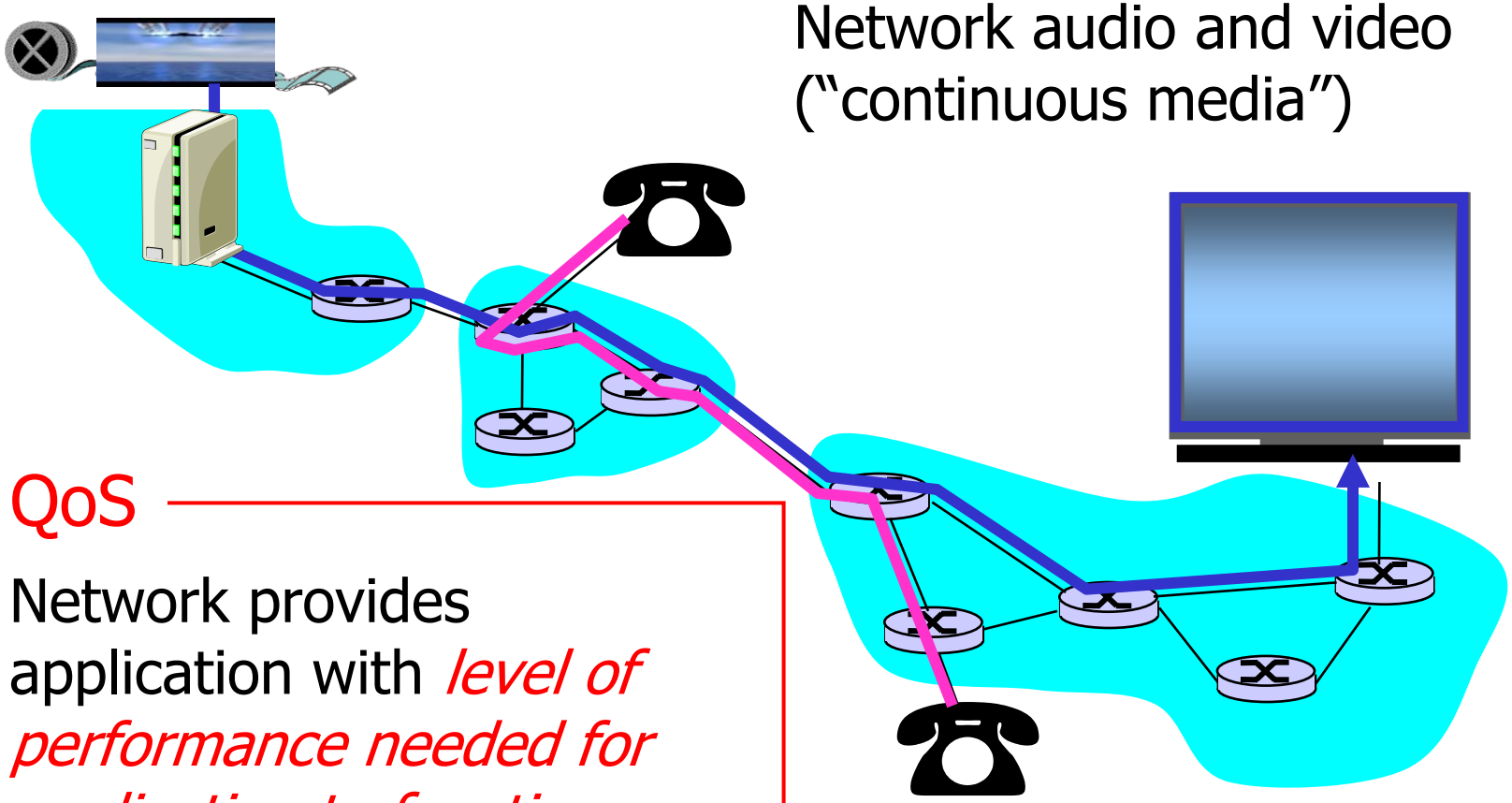
Multimedia Networking

Multimedia, Quality of Service (QoS): What is it?

Multimedia applications:
Network audio and video
("continuous media")



QoS

Network provides application with *level of performance needed for application to function.*

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

Outline

- Multimedia networking applications
- Streaming stored audio and video
- Real-time Multimedia: Internet phone study
- Protocols for Real-Time interactive applications
  - RTP, RTCP, SIP
- Beyond Best Effort
- Scheduling and Policing Mechanisms
- Integrated Services and Differentiated Services
- RSVP

MM networking applications

Classes of MM applications:

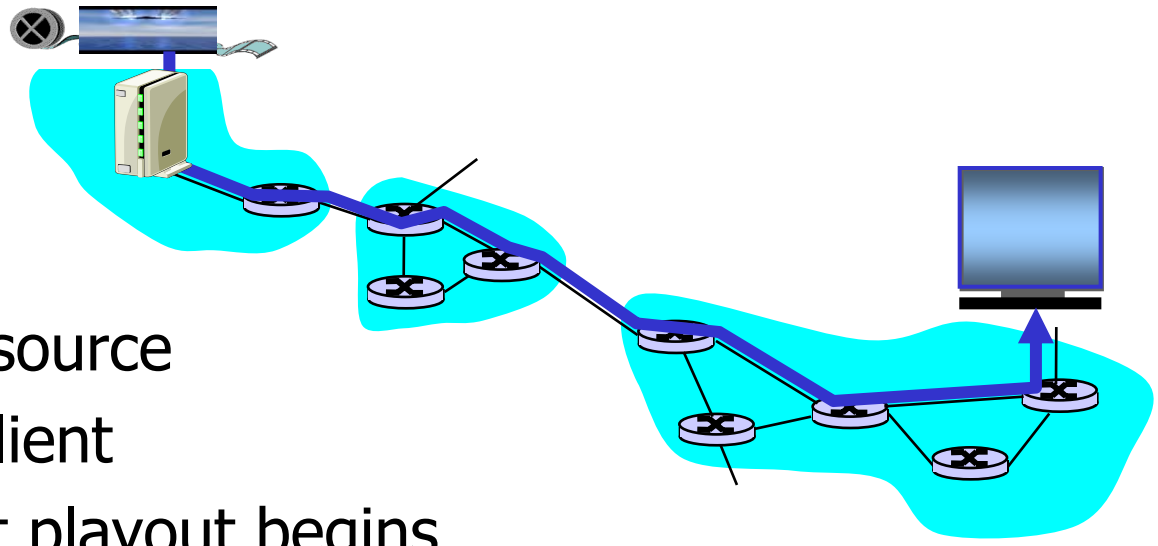
1) Streaming stored audio and video
2) Streaming live audio and video
3) Real-time interactive audio and video

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.

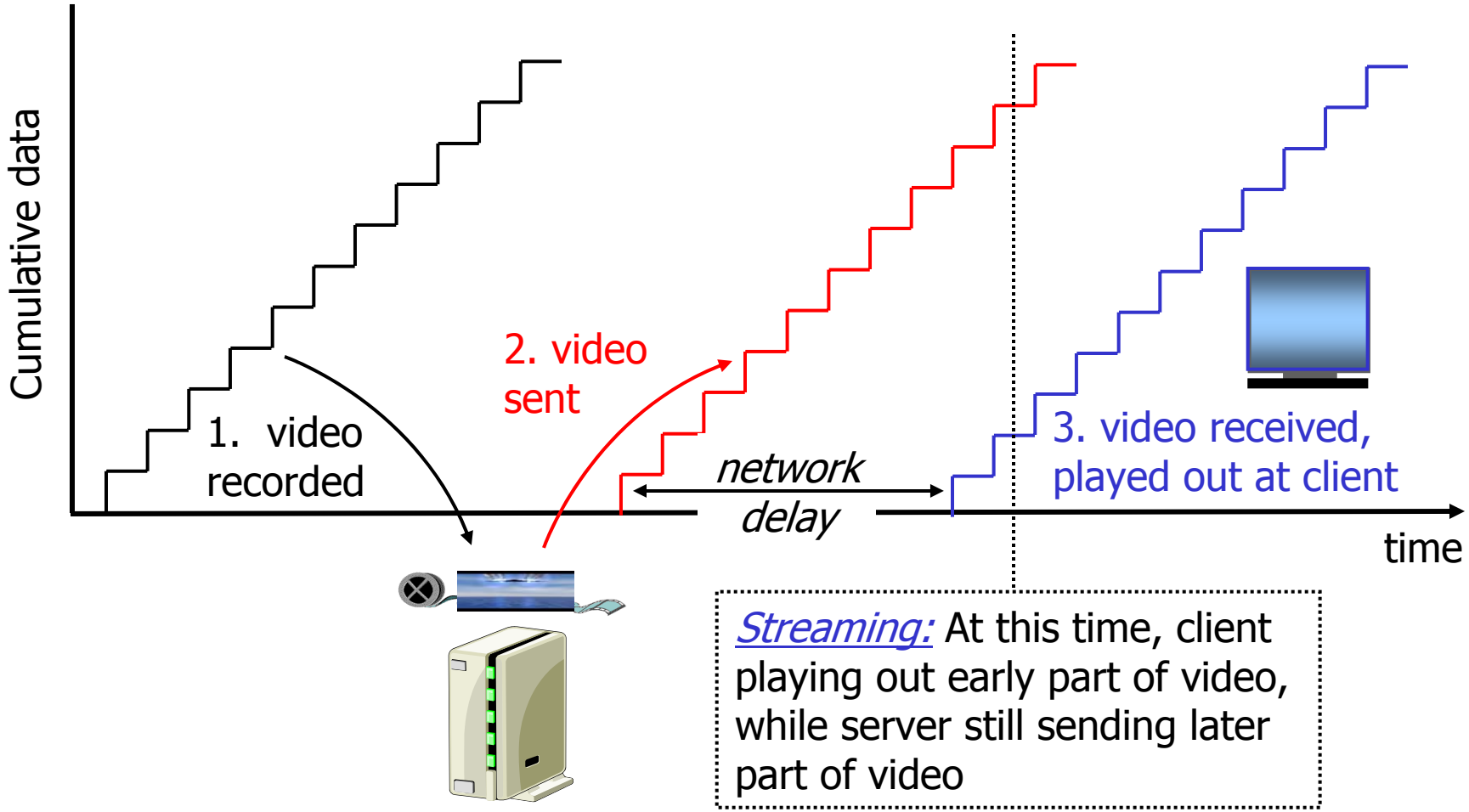
Streaming stored multimedia



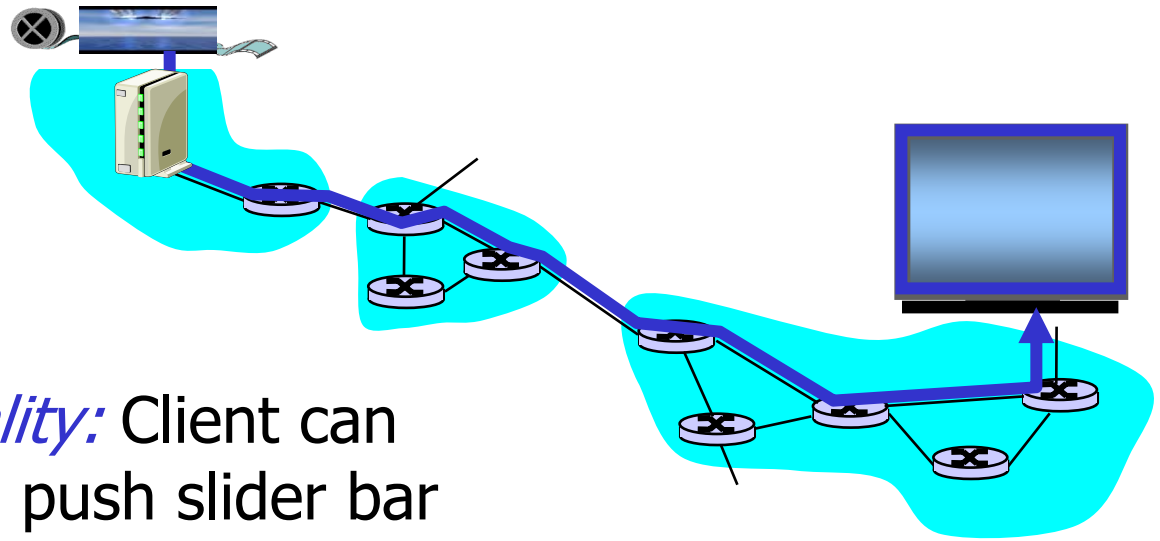
Streaming:

- 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?



Streaming stored multimedia: Interactivity



- *VCR-like functionality:* Client can pause, rewind, FF, push slider bar
  - 10 sec initial delay OK
  - 1-2 sec until command effect OK
  - RTSP often used (more later)
- Timing constraint for still-to-be transmitted data: In time for playout

Streaming live multimedia

Examples:

- Internet radio talk show
- Live sporting event

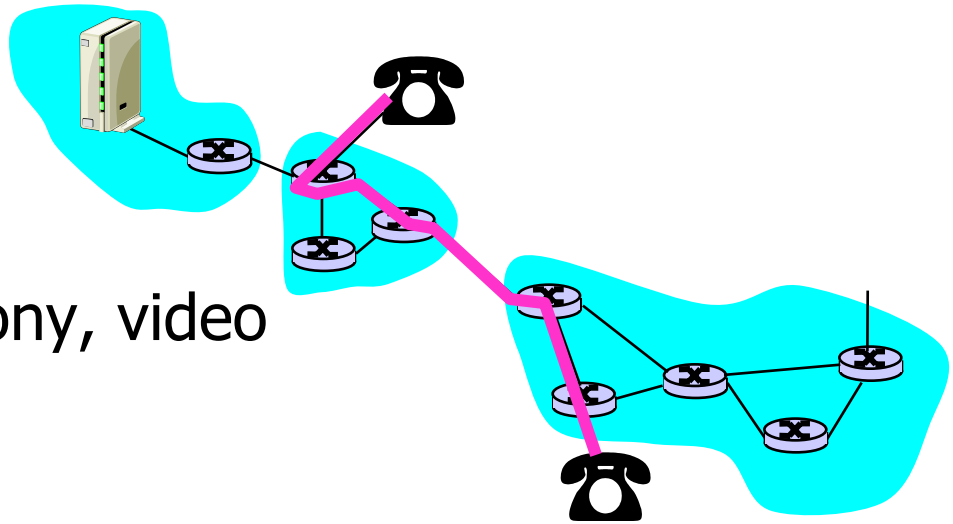
Streaming

- 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-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 "QoS"

- *No* guarantees on delay, loss

? ? ? ? ? ? ? ? ? ? ?

But you said multimedia apps requires 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 resources including bandwidth
- Requires new, complex software in hosts & routers

Laissez-faire:

- No major changes
- More bandwidth when needed
- Content distribution, application-layer multicast
  - Application layer

Differentiated services philosophy:

- Fewer changes to Internet infrastructure, yet provide 1st and 2nd class service

What's your opinion?

A few words about video compression

- Video is sequence of images displayed at constant rate
  - E.g. 24 images/sec
- Digital image is array of pixels
- Each pixel represented by bits
- Redundancy
  - Spatial
  - Temporal

Examples:

- MPEG 1 (VCD) 1.5 Mbps
- MPEG2 (DVD) 3-6 Mbps
- MPEG4 (often used in Internet, < 1 Mbps)

Research:

- Layered (scalable) video
  - Adapt layers to available bandwidth

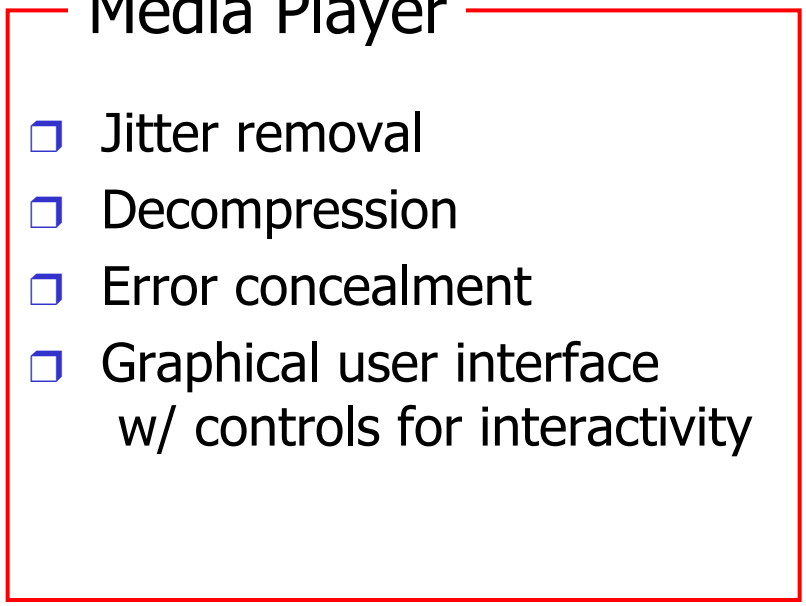
Outline

- Multimedia Networking Applications
- Streaming stored audio and video
- Real-time Multimedia: Internet phone study
- Protocols for Real-Time interactive applications
  - RTP, RTCP, SIP
- Beyond Best Effort
- Scheduling and Policing Mechanisms
- Integrated Services and Differentiated Services
- RSVP

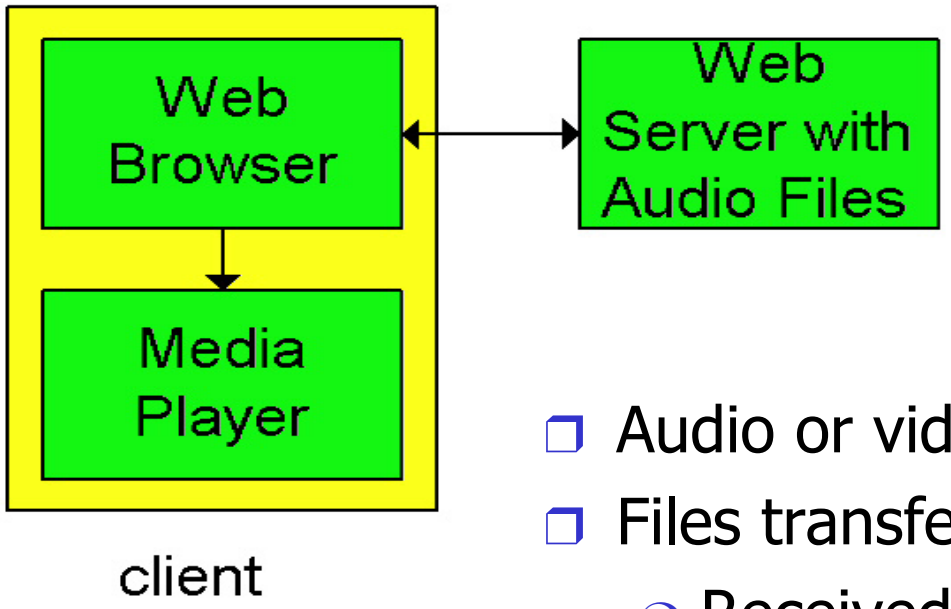
Streaming stored multimedia

Application-level streaming techniques for making the best out of best effort service:

- Client side buffering
- UDP versus TCP
- Multiple encodings of multimedia



Internet multimedia: Simplest approach

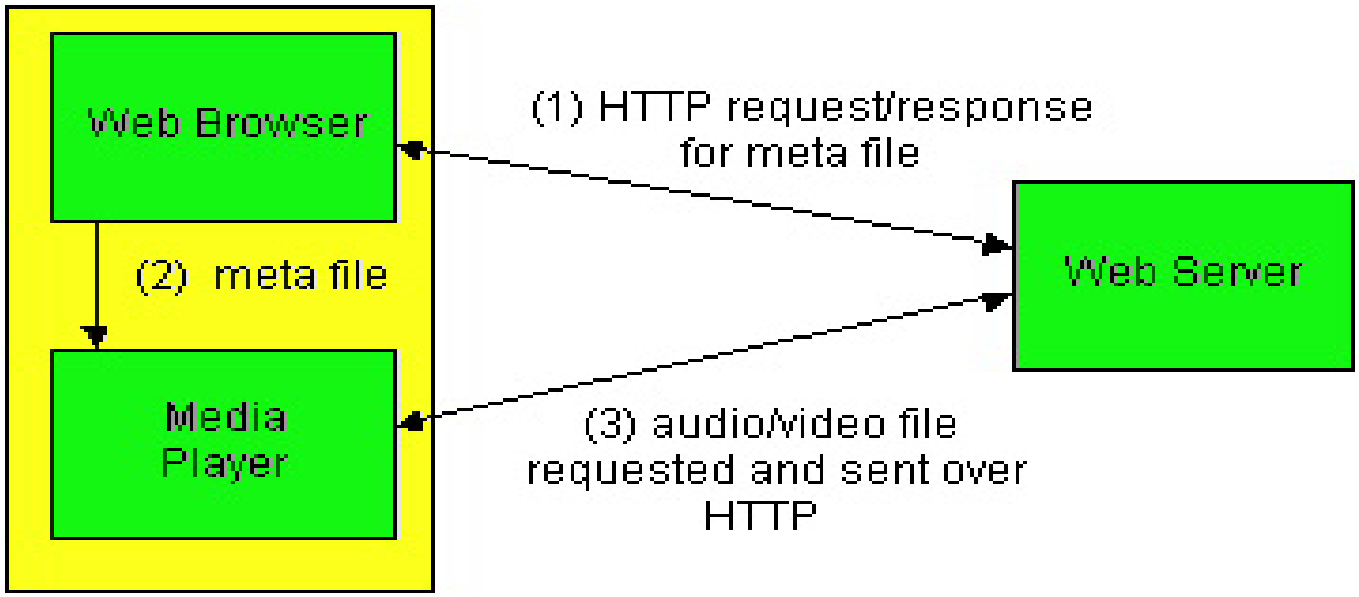


- Audio or video stored in file
- Files transferred as HTTP object
  - Received in entirety at client
  - Then passed to player

Audio, video not streamed:

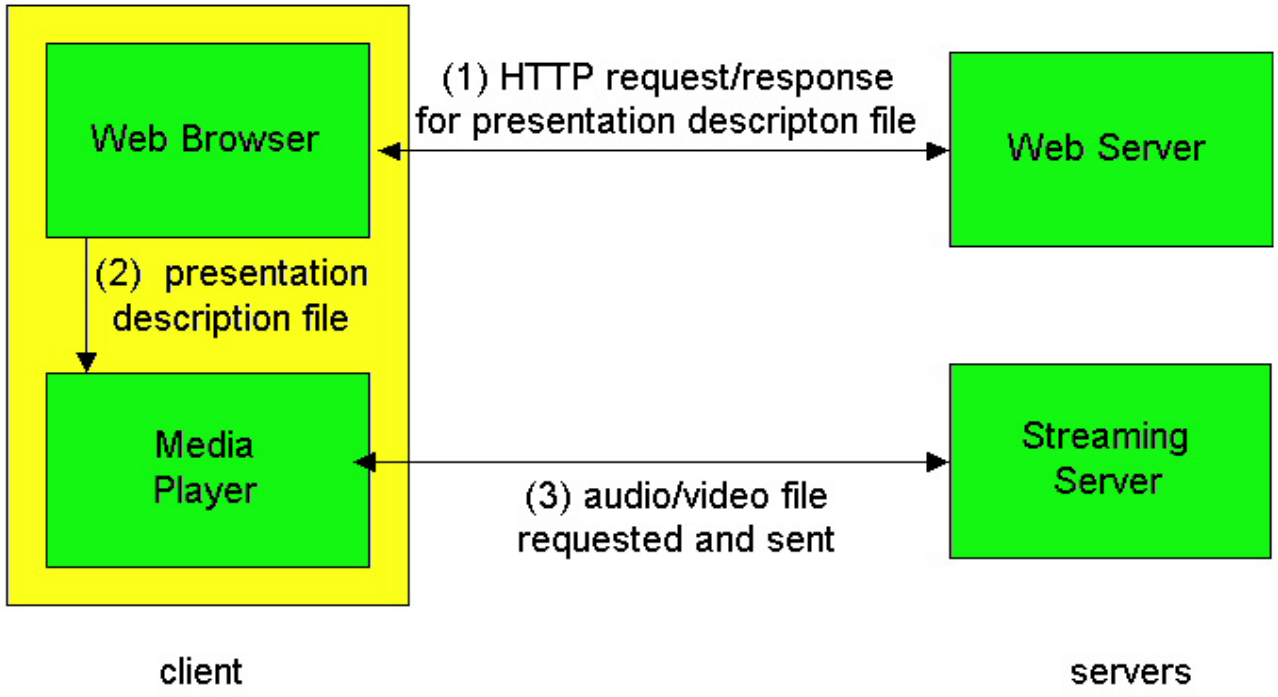
- No, "pipelining," long delays until playout!

Internet multimedia: Streaming approach



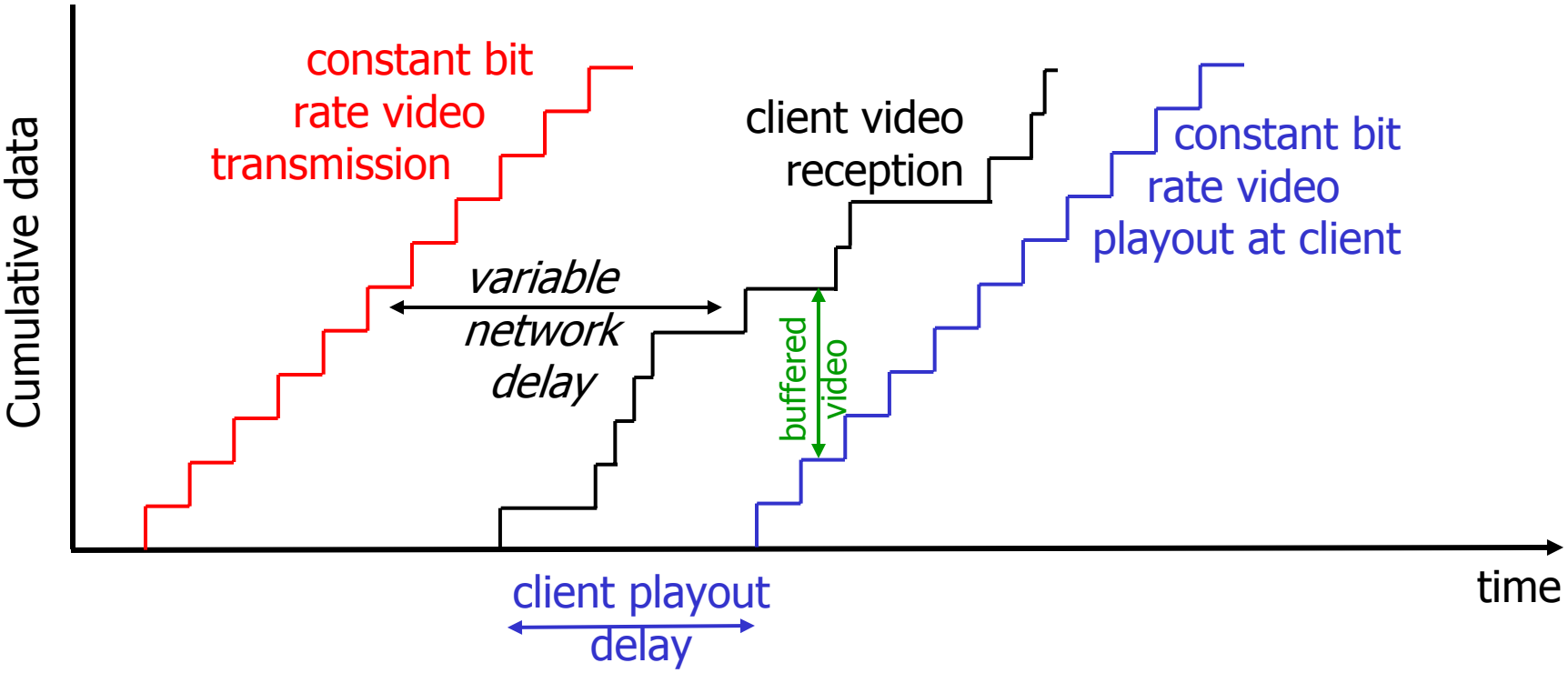
- Browser GETs **metafile**
- Browser launches player, passing metafile
- Player contacts server
- Server streams audio/video to player

Streaming from a streaming server



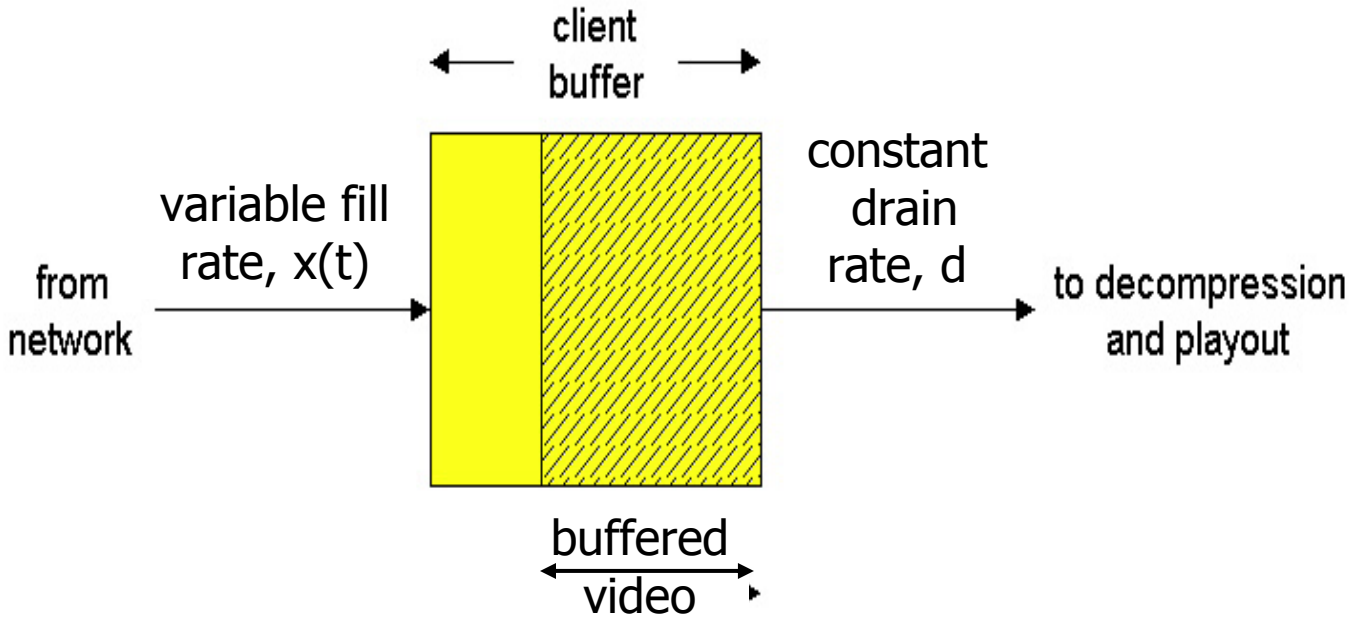
- This architecture allows for non-HTTP protocol between server and media player
- Can also use UDP instead of TCP.

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?

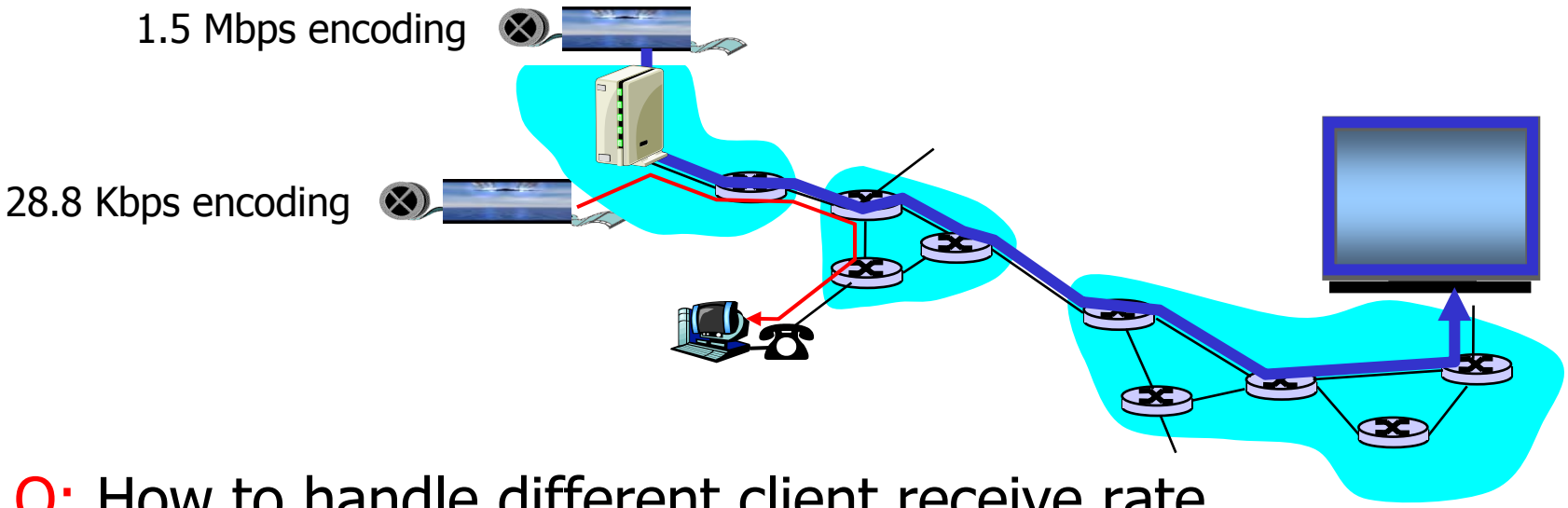
UDP

- 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
- Fill rate fluctuates due to TCP congestion control
- 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
- 100 Mbps Ethernet

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

User control of streaming media: 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
- Signaling protocol only (Transport protocol typically RTP)
- For user to control replay: 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: Out of band control

FTP uses an "out-of-band" control channel:

- A file is transferred over one TCP connection.
- Control information (directory changes, file deletion, file renaming, etc.) is sent over a separate TCP connection.
- The "out-of-band" and "in-band" channels use different port numbers.

RTSP messages are also sent out-of-band:

- RTSP control messages use different port numbers than the media stream: out-of-band.
  - Port 554
- The media stream is considered "in-band".

RTSP: Example

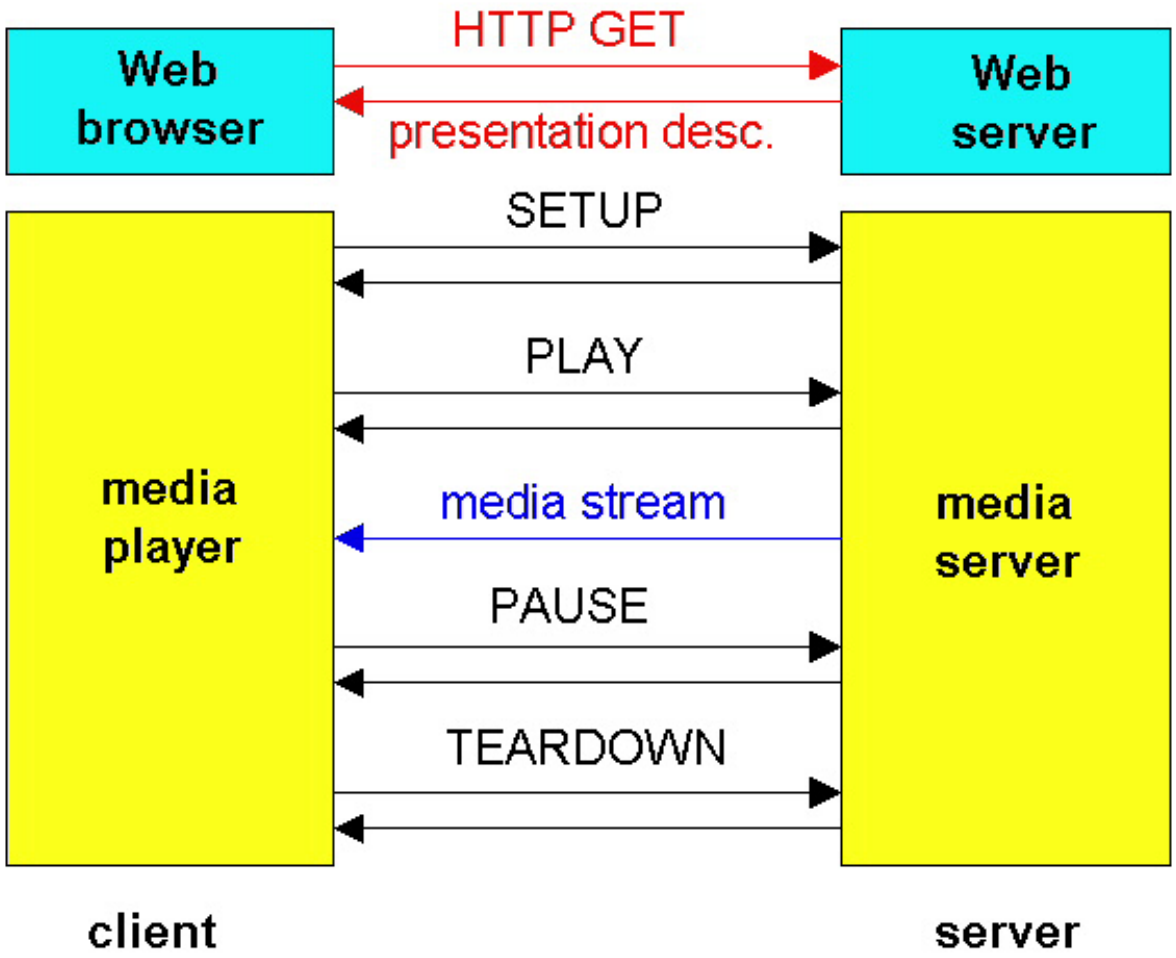
Scenario:

- 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
- Real-time Multimedia: Internet phone study
- Protocols for Real-Time interactive applications
  - RTP, RTCP, SIP
- Beyond Best Effort
- Scheduling and Policing Mechanisms
- Integrated Services and Differentiated Services
- RSVP

Real-time interactive applications

- PC-to-PC telephony
  - Instant messaging services are providing this
- PC-to-phone
  - Dialpad
  - Net2phone
  - Sipgate
- Videoconference with Webcams

Going to now look at a PC-to-PC Internet telephony example in detail

Interactive multimedia: Internet phone

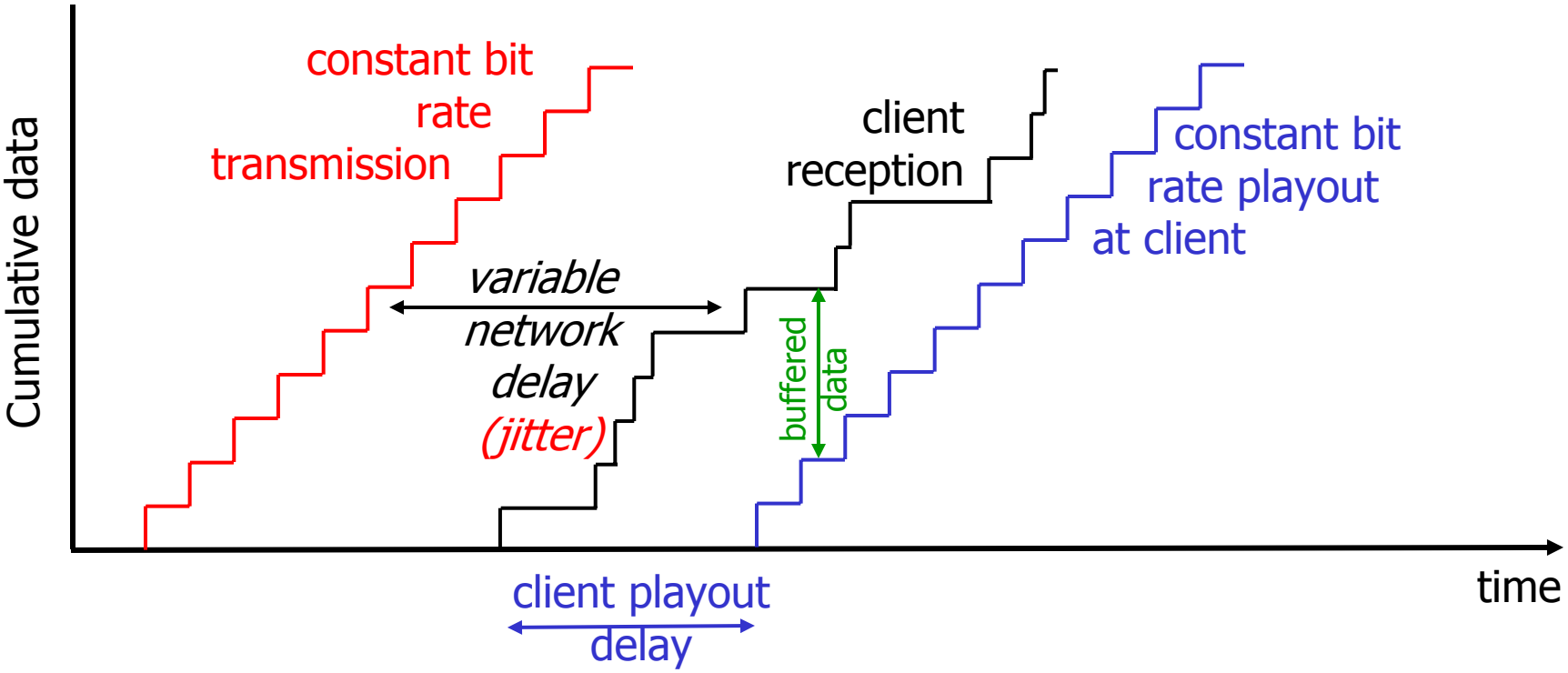
Introduce Internet telephony by way of an example

- Speaker's audio: alternating talk spurts, silent periods.
  - E.g., 64 kbps during talk spurt
- Pkts generated only during talk spurts
  - E.g., 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 talkspurt.

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



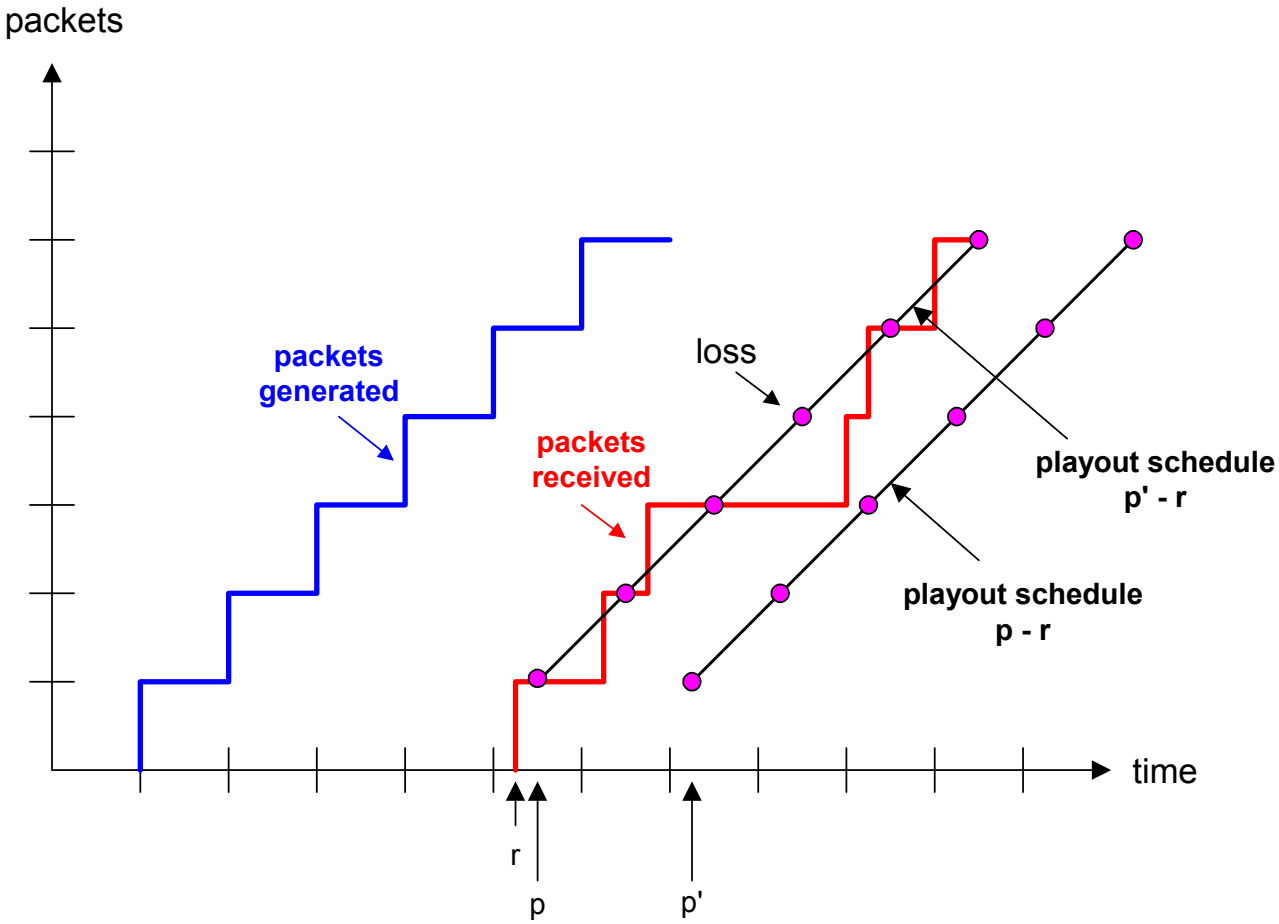
- Consider the end-to-end delays of two consecutive packets: difference can be more or less than 20 msec

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’



Adaptive playout delay

- Goal: Minimize playout delay, keeping late loss rate low
- Approach: Adaptive playout delay adjustment:
  - Estimate network delay, adjust playout delay at beginning of each talk spurt.
  - Silent periods compressed and elongated.
  - Chunks still played out every 20 msec during talk spurt.

t_i = timestamp of the ith packet

r_i = the time packet i is received by receiver

p_i = the time packet i is played at receiver

$r_i - t_i$ = network delay for ith packet

d_i = estimate of average network delay after receiving ith packet

Dynamic estimate of average delay at receiver:

$$d_i = (1-u)d_{i-1} + u(r_i - t_i)$$

where u is a fixed constant (e.g., $u = .01$).

Adaptive playout delay (2.)

Q: How does receiver determine whether packet is first in a talkspurt?

- If no loss, receiver looks at successive timestamps.
  - Difference of successive stamps > 20 msec → talk spurt begins.
- With loss possible, receiver must look at both time stamps and sequence numbers.
  - Difference of successive stamps > 20 msec and sequence numbers without gaps → talk spurt begins.

Recovery from packet loss (1)

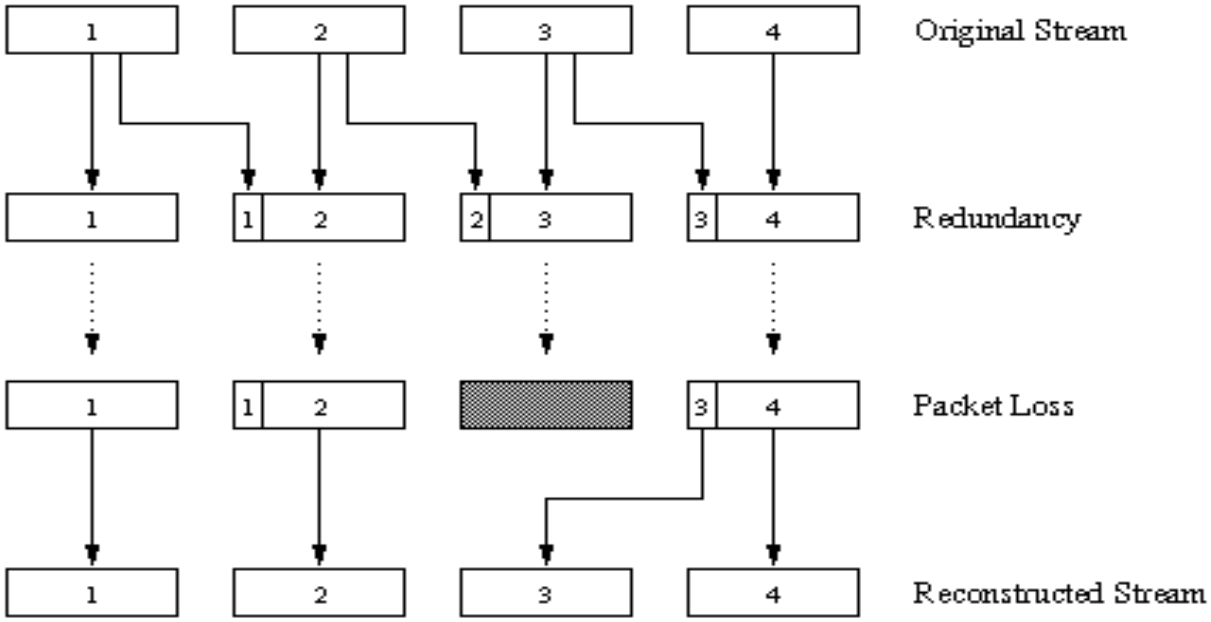
Forward error correction (FEC): simple scheme

- For every group of n chunks create a redundant chunk by exclusive OR-ing the n original chunks
- Send out n+1 chunks, increasing the bandwidth by factor 1/n.
- Can reconstruct the original n chunks if there is at most one lost chunk from the n+1 chunks
- Playout delay needs to be fixed to the time to receive all n+1 packets
- Tradeoff:
  - Increase n, less bandwidth waste
  - Increase n, longer playout delay
  - Increase n, higher probability that 2 or more chunks will be lost

Recovery from packet loss (2)

2nd FEC scheme

- "piggyback lower quality stream"
- send lower resolution audio stream as the redundant information
- for example, nominal stream PCM at 64 kbps and redundant stream GSM at 13 kbps.



- 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

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
  - Retransmissions, time permitting
  - Conceal errors: Repeat nearby data

Outline

- Multimedia Networking Applications
- Streaming stored audio and video
- Real-time Multimedia: Internet phone study
- Protocols for Real-Time interactive applications
  - RTP, RTCP, SIP
- Beyond Best Effort
- Scheduling and Policing Mechanisms
- Integrated Services and Differentiated Services
- RSVP