Multimedia Communications
@CS.NCTU

Lecture 11: Multimedia Networking

Instructor: Kate Ching-Ju Lin (林靖茹)

Why "Multimedia" Networking Matters?

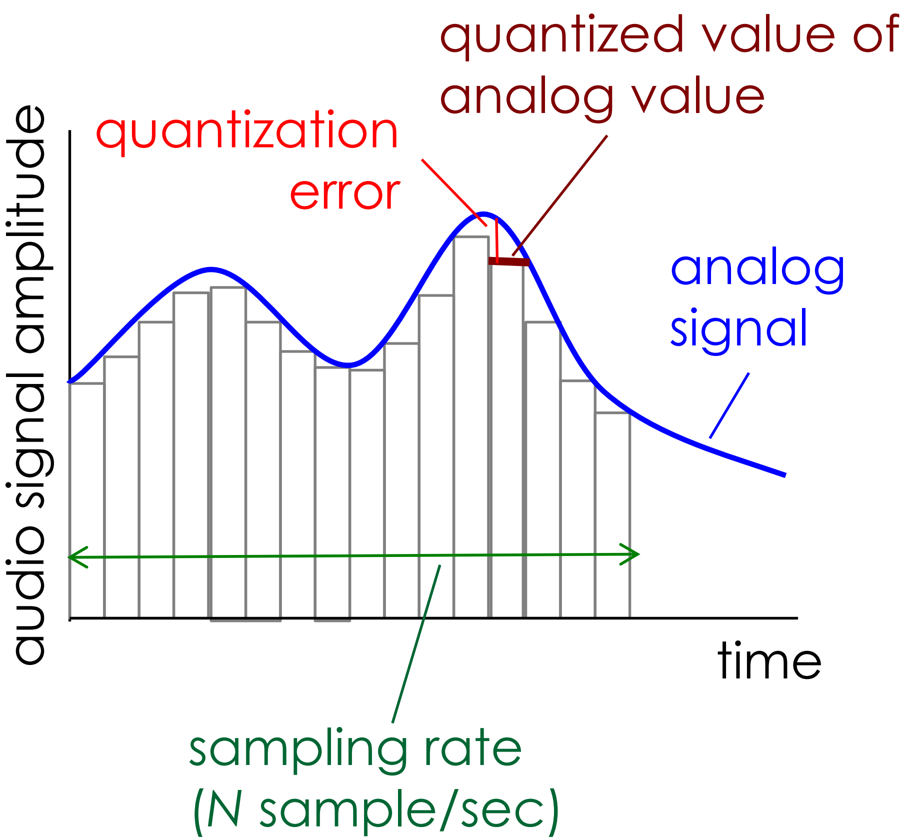
- Watching video over Internet
- Uploading user-generated content
- Telephone calls over Internet

Outline

- **Multimedia networking applications**
- Voice over IP
  - Skype
- Protocols for real-time conversational applications
  - RTP
  - SIP
- Network support for multimedia
  - Can the network (instead of application) provide mechanisms to support multimedia content delivery

Multimedia: Audio

- Analog audio signal sampled at constant rate
  - telephone: 8,000 samples/sec
  - CD music: 44,100 samples/sec
- Each sample quantized, i.e., rounded
  - e.g., 2^8=256 possible quantized values
  - each quantized value represented by bits, e.g., 8 bits for 256 values



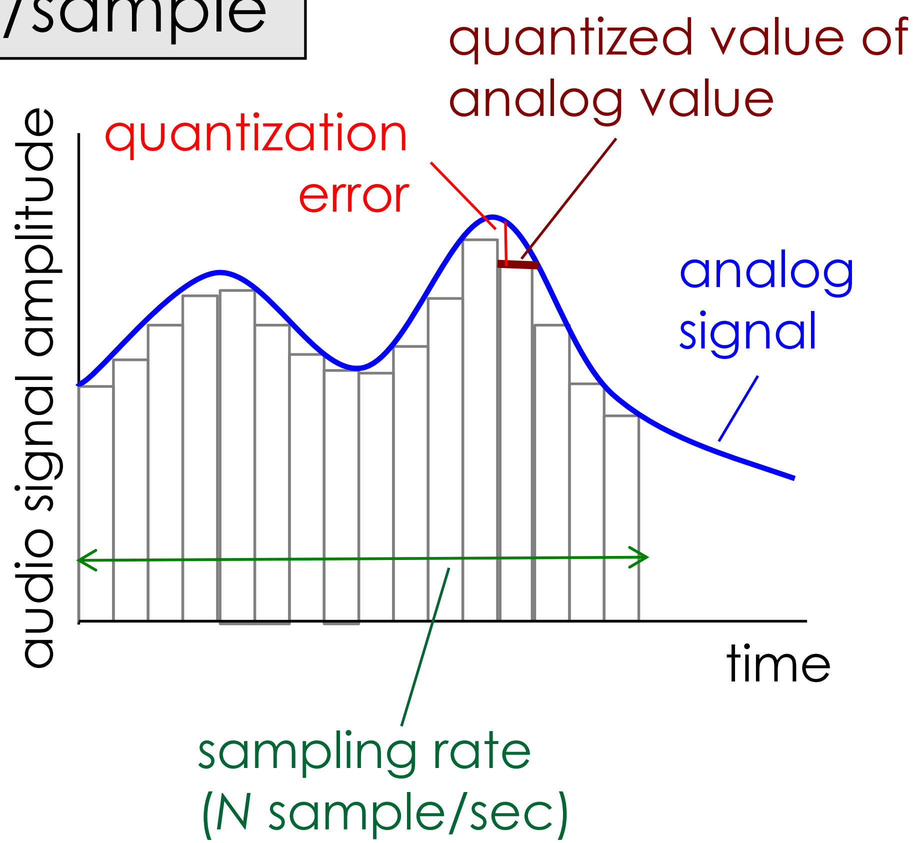
Multimedia: Audio

Audio rate = samples/sec * bits/sample

- example: 8,000 samples/sec, 256 quantized values: 64,000 bps
- receiver converts bits back to analog signal:
  - lead to quality reduction

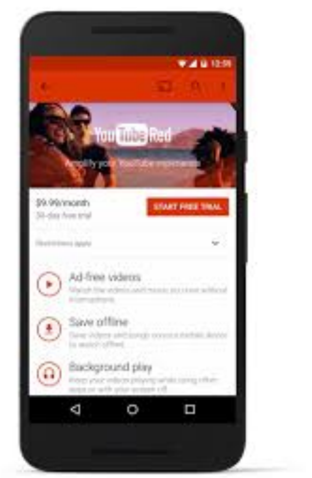
example rates

- CD: 1.411 Mbps (MPEG1 layer 3)
- MP3: 96, 128, 160 kbps
- Internet telephony: 5.3 kbps and up



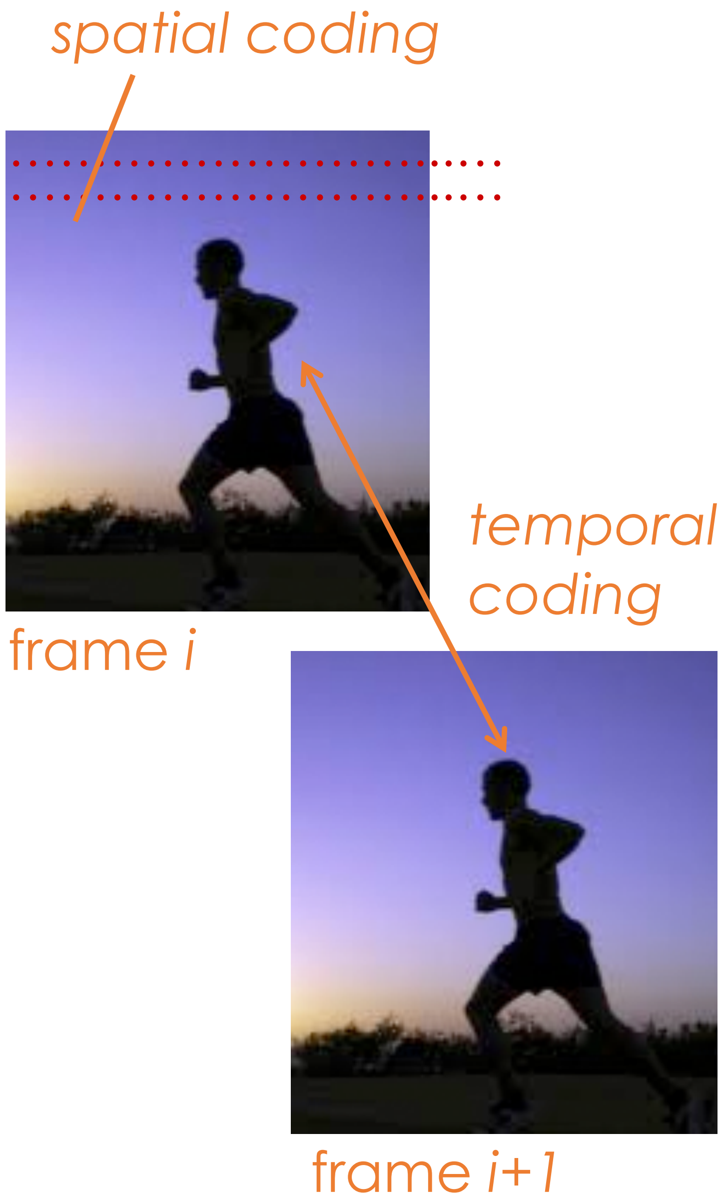
Multimedia: Video

- Data rate ranges from 100 kbps to >3Mbps
  - e.g., short chips on Facebook or Instagram
  - e.g., HD movies from iTune
- A video source can be compressed to multiple versions at different rates, e.g., 300 kbps, 1 Mbps, and 3Mbps



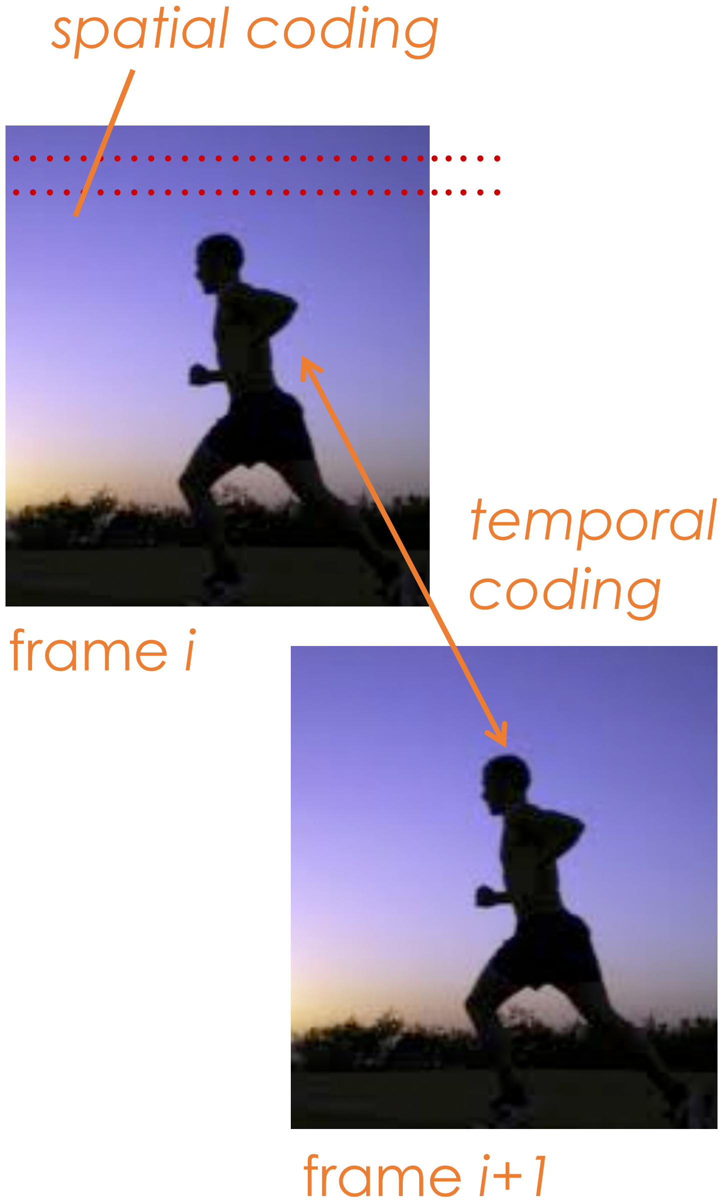
Multimedia: Video

- Video: sequence of images displayed at constant rate
  - e.g., 24 images/sec
- Digital image: array of pixels
- Coding: leverage redundancy *within* and *between*
  - spatial (within image)
  - temporal (from one image to next)



Multimedia: Video

- **CBR: (constant bit rate)**: video encoding rate fixed
- **VBR: (variable bit rate)**: video encoding rate changes as amount of spatial, temporal coding changes
- **Examples**:
  - MPEG 1 (CD-ROM) 1.5 Mbps
  - MPEG2 (DVD) 3-6 Mbps
  - MPEG4 (often used in Internet, < 1 Mbps)



Multimedia Application Types

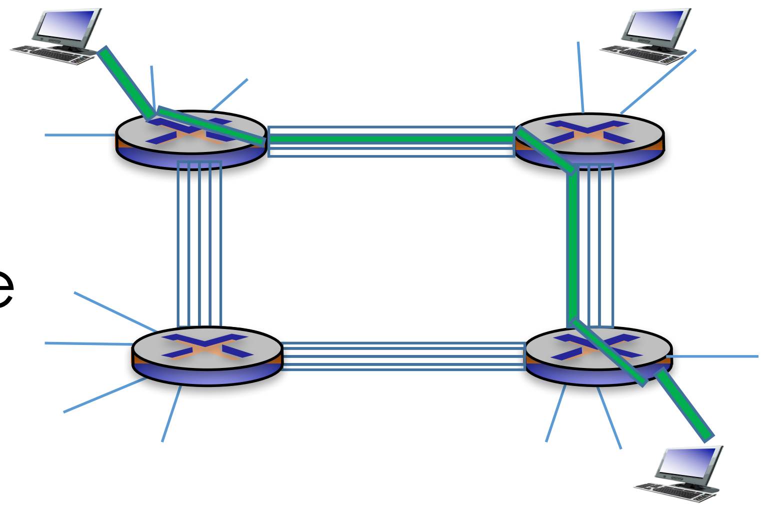
- **Streaming, stored** audio, video
  - streaming: can begin playout before downloading entire file
  - stored (at server): can transmit faster than audio/video will be rendered (implies storing/buffering at client)
  - requested by client on demand
  - e.g., YouTube, Netflix, Hulu, occupying >50% of Internet traffic
- **Conversational** voice/video over IP
  - interactive nature of human-to-human conversation limits delay tolerance, e.g., Skype, Google handout
  - highly delay-sensitive, but loss-tolerant
- **Streaming live** audio, video
  - e.g., live sporting event (broadcasting)

Outline

- Multimedia networking applications
- **Voice over IP**
  - Skype
- Protocols for real-time conversational applications
  - RTP
  - SIP
- Network support for multimedia
  - Can the network (instead of application) provide mechanisms to support multimedia content delivery

Voice-over-IP (VoIP)

- Real-time conversational voice, often known as **Internet telephony**
- Delay sensitive
  - higher delays noticeable, impair interactivity
  - < 150 msec: good
  - > 400 msec: bad
- Loss tolerant
  - A few losses only loss only causes occasional glitches
- **Goal:** similar to traditional circuit-switched telephone service
  - performance guarantee



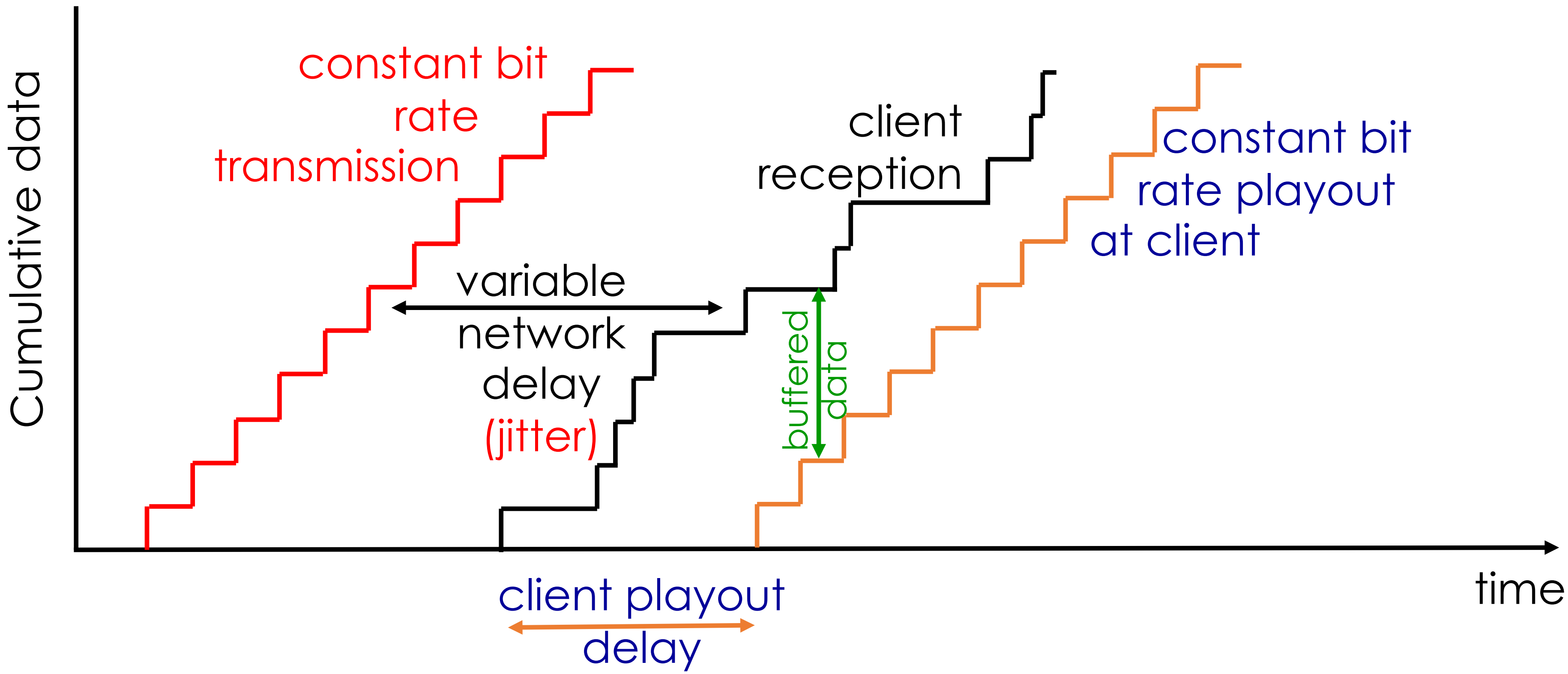
VoIP Characteristics

- Internet: provide best-effort services, no performance guarantee
- Application-layer solution
  - Bit-streams are partitioned into chunks
  - 20 msec chunks at 8 Kbytes/sec: 160 bytes of data
- Chunk+header encapsulated into UDP or TCP segments
- At sender, application sends segments into socket every 20 msec
- At receiver, determine (1) when to play back a chunk, and (2) how to deal with losses
  - Playing back chunks immediately as they arrive might not be a good strategy

VoIP: Packet Loss, Delay

- Network loss: IP datagram (UDP) lost due to network congestion (router buffer overflow)
  - TCP prevents losses, but retransmissions increases delay
- 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
  - packets arrived too late are effectively lost
- Loss tolerance: packet loss rates between 1% and 10% can be tolerated
  - depending on voice encoding, loss concealment

Delay Jitter



- End-to-end delays of two consecutive packets: difference can be more or less than 20 msec (transmission time difference)

VoIP: Fixed Playout Delay

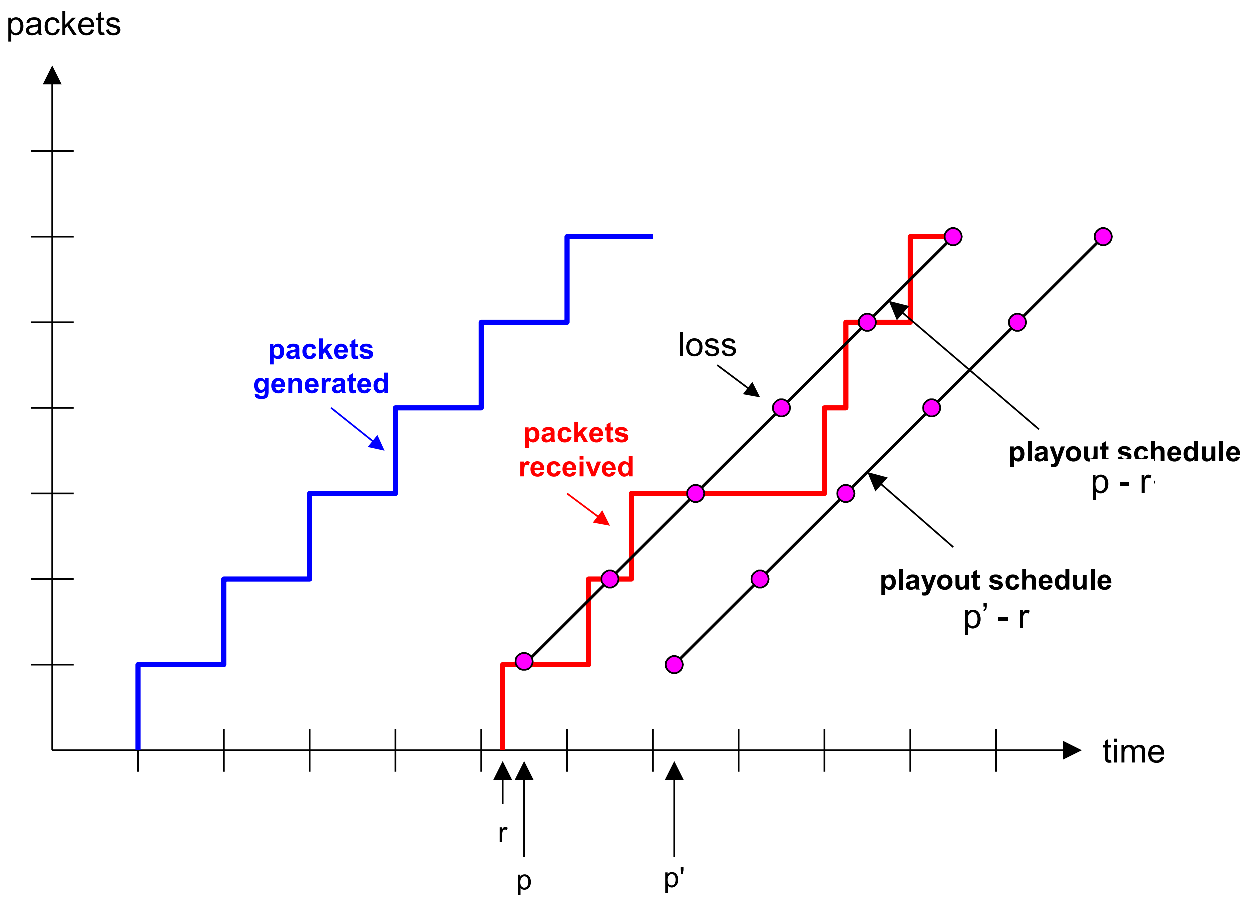
- Leverage sequence number and time-stamp

- 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 in choosing *q*:
  - large q: less packet loss, longer startup latency
  - small q: better interactive experience

VoIP: 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:* low playout delay, low late loss rate
- *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
- Adaptively estimate packet delay: (EWMA - exponentially weighted moving average, recall TCP RTT estimate):

$$d_i = (1-\alpha)d_{i-1} + \alpha\,(r_i - t_i)$$

- d_i: *delay estimate after ith packet*
- α: *small constant, e.g. 0.1*
- $r_i - t_i$: *time received - time sent (timestamp)* = *measured delay of ith packet*

Adaptive Playout Delay

- Also useful to estimate average deviation of delay, v_i:

$$v_i = (1-b)v_{i-1} + b \; |r_i - t_i - d_i|$$

- Estimates d_i, v_i calculated for every received packet, but used only at start of talk spurt

- For first packet in talk spurt, playout time is:

$$playout\text{-}time_i = t_i + d_i + Kv_i$$

(constant, e.g., 4)

- Remaining packets in talkspurt are played out periodically

VoIP: Recovery From Packet Loss

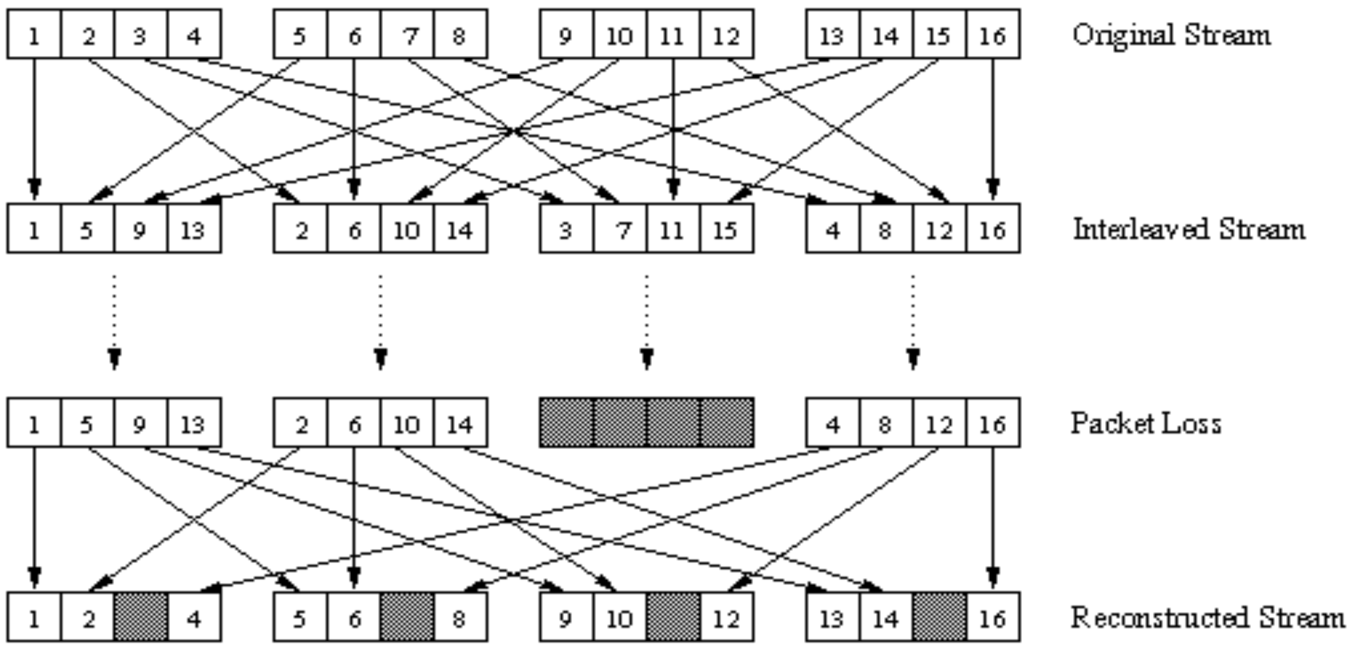
Challenge: recover from packet loss given small tolerable delay between original transmission and playout

- Each ACK/NAK takes ~ one RTT → need longer delay
- Alternative: Forward Error Correction (FEC)
  - send enough bits to allow recovery without retransmission (See Ch. 5)

Simple FEC

- For every group of *n* chunks, create redundant chunk by exclusive OR-ing *n* original chunks
- Send *n+1* chunks, increasing bandwidth by factor *1/n*
- Can reconstruct original *n* chunks if at most one lost chunk from *n+1* chunks, with playout delay

VoIP: Recovery From Packet Loss



Interleaving to conceal burst loss:

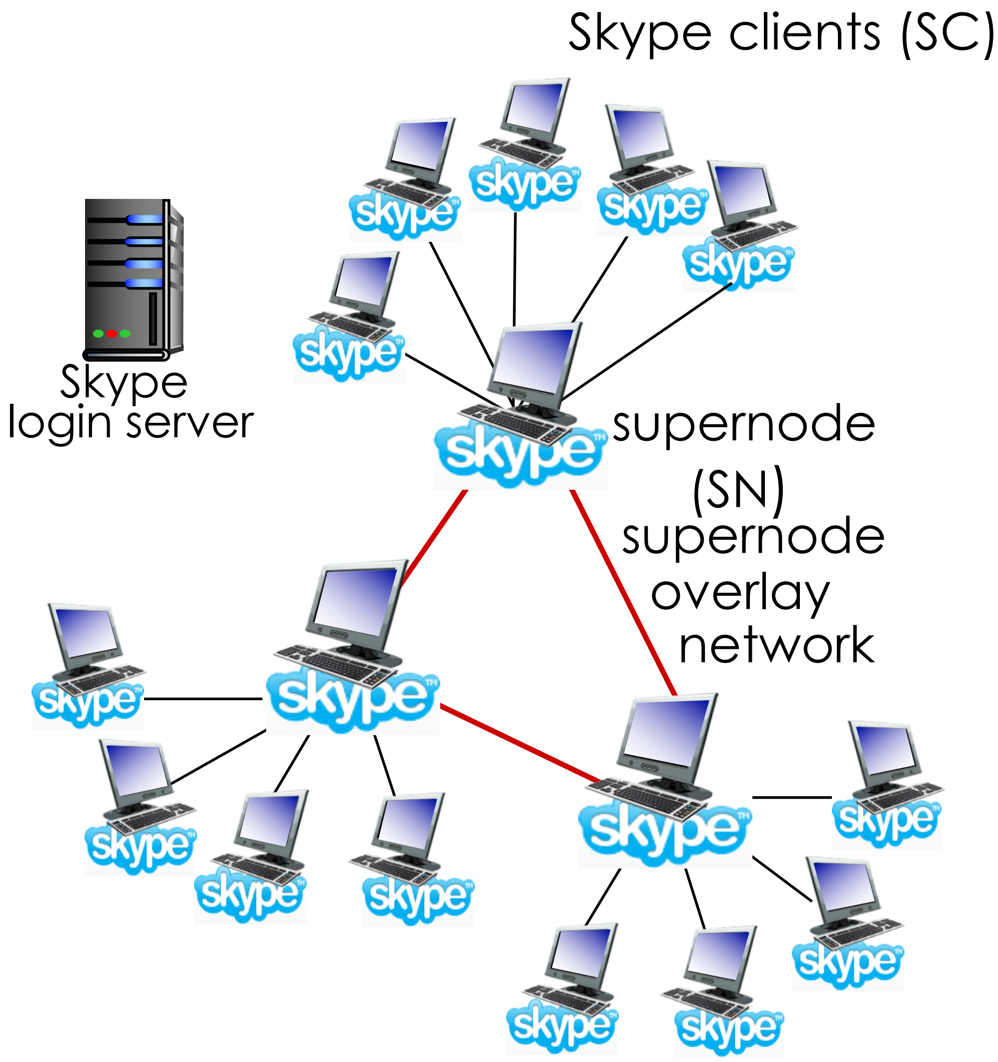
- Audio chunks divided into smaller units and shuffle the small units
- If packet lost, still have most of every original chunk
  - Work with FEC
- No redundancy overhead, but increases playout delay

Voice-over-IP: Skype

- Acquired by Microsoft in 2011 for over $8 billion
- Proprietary protocol, packets are encrypted → hard to trace their operations
- Audio and video packets are sent over UDP
- Control packets are sent over TCP
- Exploit P2P structure
- Deal with the NAT problem via relays

Voice-over-IP: Skype

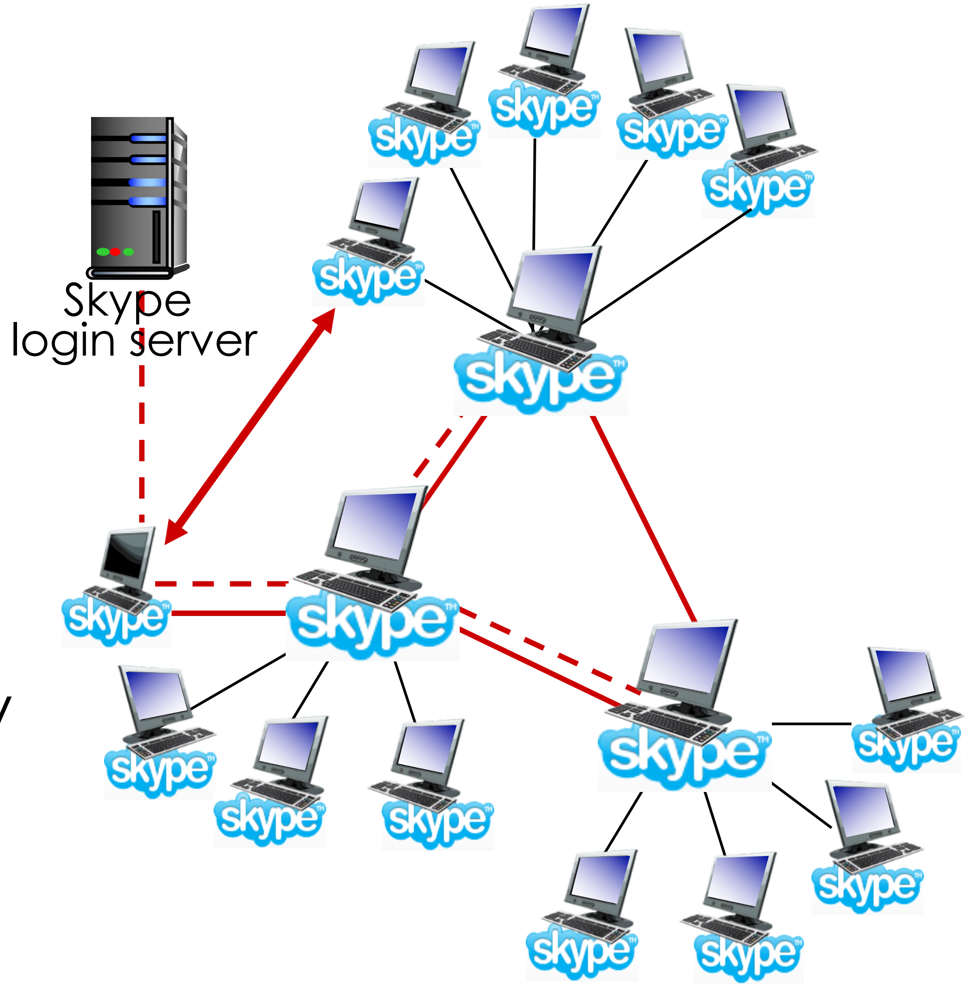
- **Proprietary** application-layer protocol (inferred via reverse engineering)
- P2P components:
  - clients: Skype peers connect directly to each other for VoIP call
  - super nodes (SN): Skype peers with special functions
  - overlay network: among SNs to locate SCs
  - login server: only for authentication



Voice-over-IP: Skype

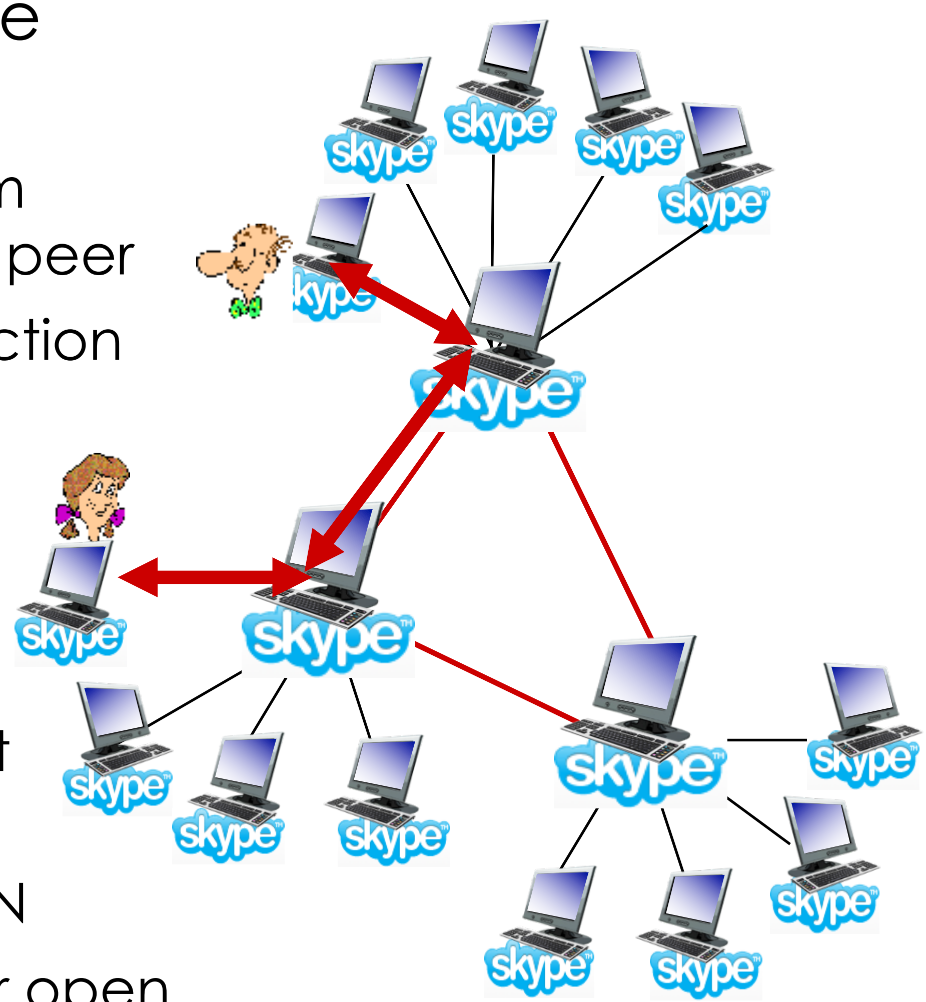
Skype client operation:

1. Joins Skype network by contacting SN (IP address cached) using TCP
2. Logs-in (username, password) to centralized Skype login server
3. Obtains IP address for callee from SN, SN overlay
   - or client buddy list
4. Initiate call directly to callee



Skype: Peers as Relays

- Problem: both Alice, Bob are behind "NATs"
  - NAT prevents outside peer from initiating connection to insider peer
  - inside peer can initiate connection to outside
- **Relay solution**: Alice, Bob maintain open connection to their SNs
  - Alice signals her SN to connect to Bob
  - Alice's SN connects to Bob's SN
  - Bob's SN connects to Bob over open connection Bob initially initiated to his SN



Outline

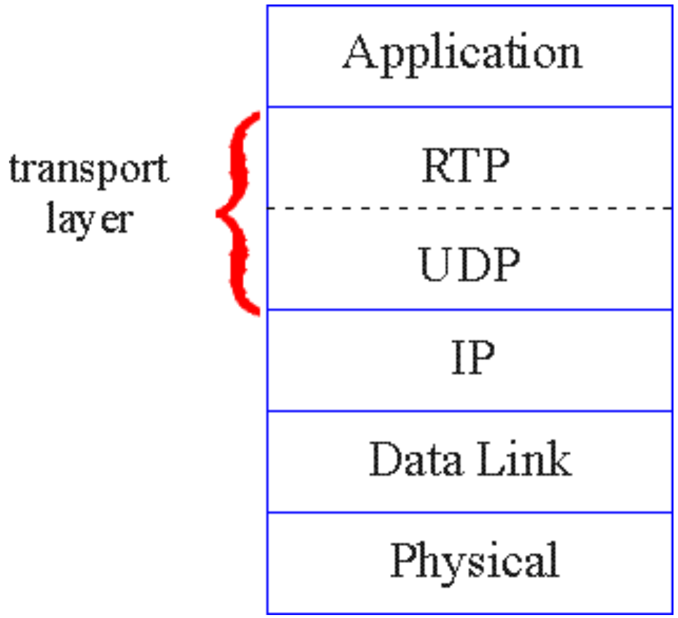
- Multimedia networking applications
- Voice over IP
  - Skype
- **Protocols for real-time conversational applications**
  - RTP
  - SIP
- Network support for multimedia
  - Can the network (instead of application) provide mechanisms to support multimedia content delivery

Real-Time Protocol (RTP)

- RTP specifies packet structure for packets carrying audio, video data
- RFC 3550
- RTP packet provides
  - payload type (audio/video)
  - packet sequence numbering
  - time stamping
- RTP runs in end systems
- RTP packets encapsulated in UDP segments
- Interoperability: if two VoIP applications run RTP, they may be able to work together
- Can be used for ACC, MP3, MPEG, H263, etc

RTP Runs on Top of UDP

- RTP libraries provide transport-layer interface that **extends UDP**:
  - port numbers, IP addresses
  - payload type identification
  - packet sequence numbering
  - time-stamping



RTP example

Example: sending 64 kbps PCM-encoded voice over RTP

- Application collects encoded data in chunks, e.g., every 20 msec = 160 bytes in a chunk
- Audio chunk + RTP header form RTP packet, which is encapsulated in UDP segment

- RTP header indicates type of audio encoding in each packet
  - sender can change encoding during conference
- RTP header also contains sequence numbers, timestamps

RTP and QoS

- RTP *does not* provide any mechanism to ensure timely data delivery or other QoS guarantees

- RTP encapsulation only seen at end systems (*not* by intermediate routers)
    - routers provide best-effort service, making no special effort to ensure that RTP packets arrive at destination in timely matter

RTP Header

payload type	*sequence number*	*time stamp*	*Synchronization Source ID*	*Miscellaneous fields*

- **Payload type (7 bits):** indicates type of encoding currently being used. If sender changes encoding during call, sender informs receiver via payload type field
  - Payload type 0: PCM mu-law, 64 kbps
  - Payload type 3: GSM, 13 kbps
  - Payload type 7: LPC, 2.4 kbps
  - Payload type 26: Motion JPEG
  - Payload type 31: H.261
  - Payload type 33: MPEG2 video
- **sequence # (16 bits):** increment by one for each RTP packet sent
  - detect packet loss, restore packet sequence

RTP Header

payload type	*sequence number*	*time stamp*	*Synchronization Source ID*	*Miscellaneous fields*

- **Timestamp field (32 bits long):** sampling instant of first byte in this RTP data packet
  - for audio, timestamp clock increments by one for each sampling period (e.g., each 125 usecs for 8 KHz sampling clock)
  - if application generates chunks of 160 encoded samples, timestamp increases by 160 for each RTP packet when source is active. Timestamp clock continues to increase at constant rate when source is inactive.

- **SSRC field (32 bits long):** ID randomly selected by the source. Each stream in RTP session has distinct SSRC

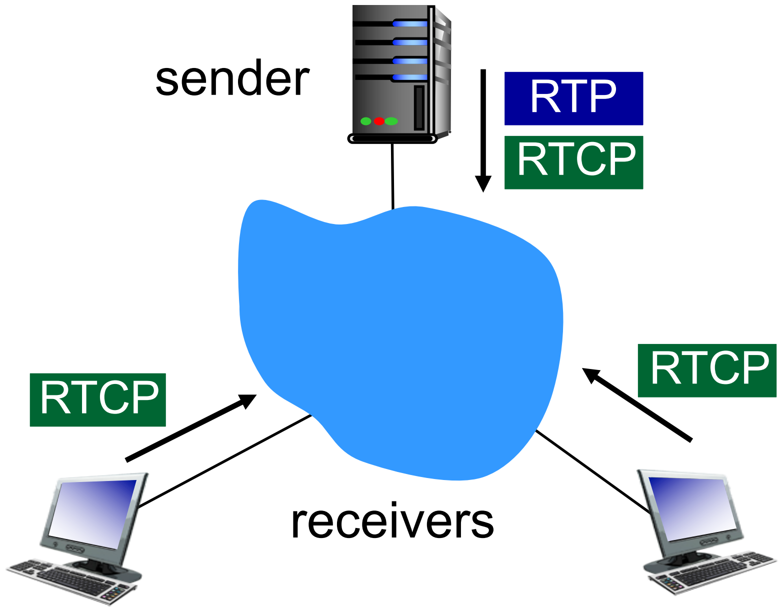
Real-Time Control Protocol (RTCP)

- Works in conjunction with RTP

- Each participant in RTP session periodically sends RTCP control packets to all other participants

- Each RTCP packet contains sender and/or receiver reports
  - report statistics useful to application: # packets sent, # packets lost, interarrival jitter
- Feedback used to control performance
  - sender may modify its transmissions based on feedback

RTCP: Multiple Multicast Senders



- Each RTP session: typically a single multicast address; all RTP /RTCP packets belonging to session use multicast address
- RTP, RTCP packets distinguished from each other via distinct port numbers
- To limit traffic, each participant reduces RTCP traffic as number of conference participants increases

RTCP: Packet Types

Receiver report packets:

- Fraction of packets lost, last sequence number, average inter-arrival jitter

Sender report packets:

- SSRC of RTP stream, current time, number of packets sent, number of bytes sent

Source description packets:

- E-mail address of sender, sender's name, SSRC of associated RTP stream
- Provide mapping between the SSRC and the user/host name

RTCP: Stream Synchronization

- RTCP can synchronize different media streams within a RTP session
- E.g., videoconferencing app: each sender generates one RTP stream for video, one for audio
- Timestamps in RTP packets tied to the video, audio sampling clocks
  - *not* tied to wall-clock time
- Each RTCP sender-report packet contains (for most recently generated packet in associated RTP stream):
  - timestamp of RTP packet
  - wall-clock time for when packet was created
- Receivers uses association to synchronize playout of audio, video

RTCP: Bandwidth Scaling

RTCP attempts to limit its traffic to 5% of session bandwidth

Example : one sender, sending video at 2 Mbps

- RTCP attempts to limit RTCP traffic to 100 Kbps
- RTCP gives 75% of rate to receivers; remaining 25% to sender
- 75 kbps is equally shared among receivers:
  - with R receivers, each receiver gets to send RTCP traffic at 75/R kbps.
- Sender gets to send RTCP traffic at 25 kbps.
- Participant determines RTCP packet transmission period by calculating avg RTCP packet size (across entire session) and dividing by allocated rate

SIP: Session Initiation Protocol [RFC 3261]

Long-term vision:

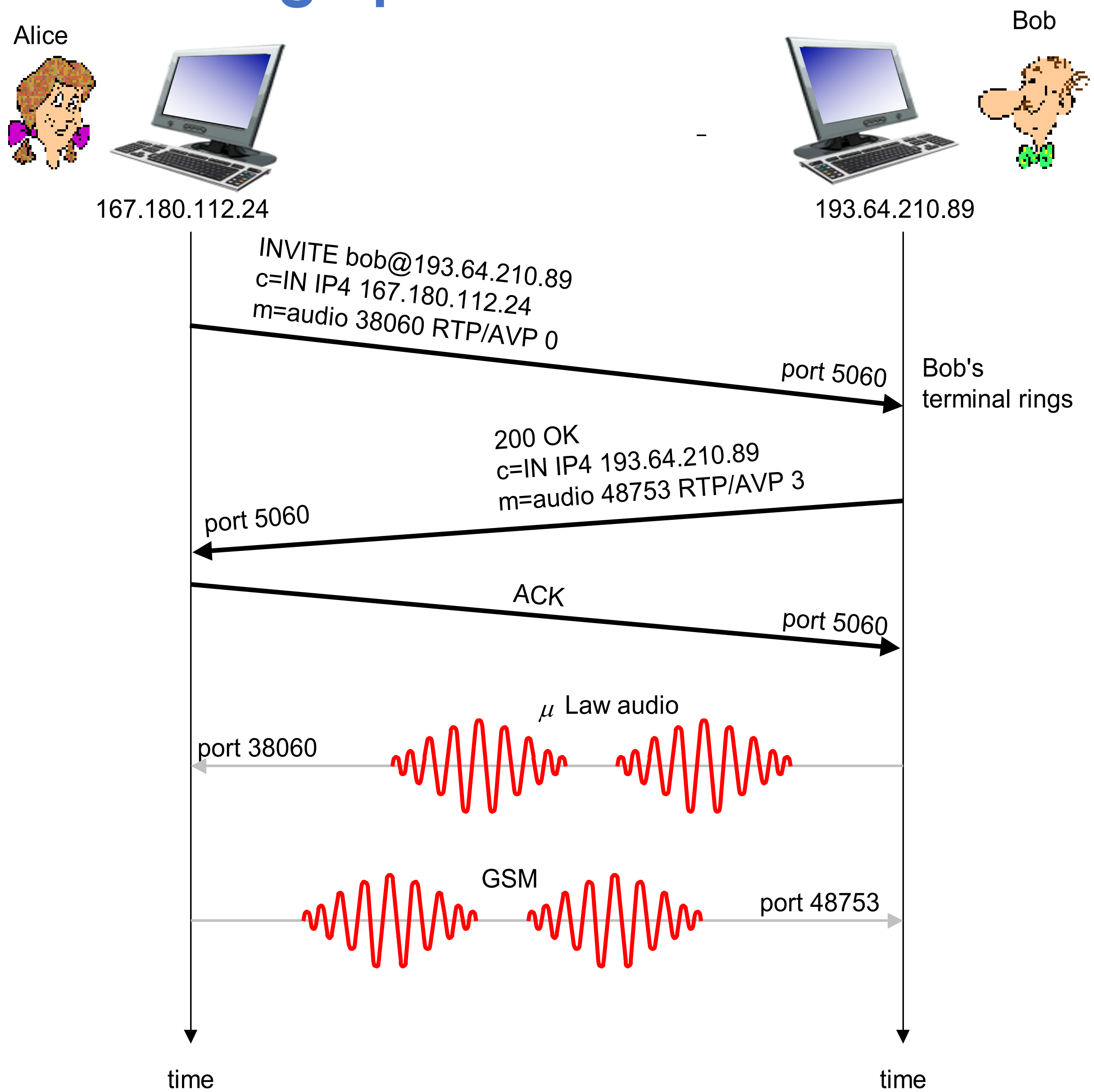
- All telephone calls, video conference calls take place over Internet
- People identified by names or e-mail addresses, rather than by phone numbers
- Can reach callee *(if callee so desires)*, no matter where callee roams, no matter what IP device callee is currently using

SIP Services

- SIP provides mechanisms for call setup:
  - for caller to let callee know she wants to establish a call
  - so caller, callee can agree on media type, encoding
  - to end call
- Determine current IP address of callee:
  - maps mnemonic identifier to current IP address
- Call management:
  - add new media streams during call
  - change encoding during call
  - invite others
  - transfer, hold calls

Example

Setting up call to known IP address



- Alice's SIP invite message indicates her port number, IP address, encoding she prefers to receive (PCM μlaw)
- Bob's 200 OK message indicates his port number, IP address, preferred encoding (GSM)
- SIP messages can be sent over TCP or UDP; here sent over RTP/UDP
- Default SIP port number is 5060

Setting Up a Call (cont.)

- Codec negotiation:
  - Suppose Bob doesn't have PCM μlaw encoder
  - Bob will instead reply with 606 Not Acceptable Reply, listing his encoders. Alice can then send new INVITE message, advertising different encoder
- Rejecting a call
  - Bob can reject with replies "busy," "gone," "payment required," "forbidden"
- Media can be sent over RTP or some other protocol

Example of SIP message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notes:

- HTTP message syntax
- sdp = session description protocol
- Call-ID is unique for every call

- Here we don't know Bob's IP address
  - intermediate SIP servers needed
- Alice sends, receives SIP messages using SIP default port 506
- Alice specifies in header that SIP client sends, receives SIP messages over UDP

Name Translation, User Location

- Caller wants to call callee, but only has callee's name or e-mail address.
- Need to get IP address of callee's current host:
  - user moves around
  - DHCP protocol
  - user has different IP devices (PC, smartphone, car device)
- Result can be based on:
  - time of day (work, home)
  - caller (don't want boss to call you at home)
  - status of callee (calls sent to voicemail when callee is already talking to someone)

SIP Registrar

- One function of SIP server: registrar
- When Bob starts SIP client, client sends SIP REGISTER message to Bob's registrar server

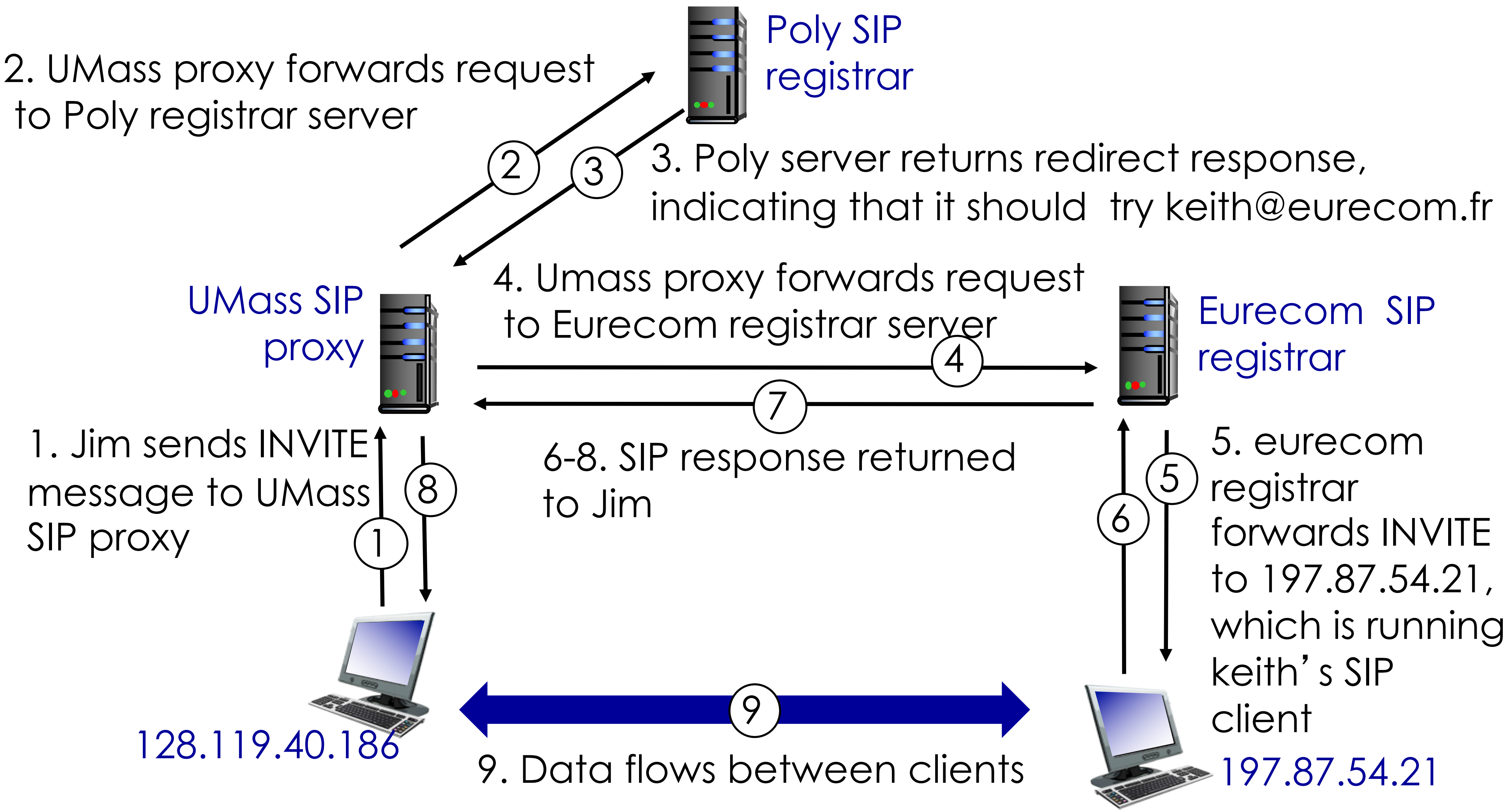
Register message:

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

SIP Proxy

- Another function of SIP server: **proxy**
- Alice sends invite message to her proxy server
  - Contains address sip:bob@domain.com
  - Proxy responsible for routing SIP messages to callee, possibly through multiple proxies
- Bob sends response back through same set of SIP proxies
- Proxy returns Bob's SIP response message to Alice
  - Contains Bob's IP address
- SIP proxy analogous to local DNS server plus TCP setup

SIP example: jim@umass.edu calls keith@poly.edu



Comparison with H.323

- H.323: another signaling protocol for real-time, interactive multimedia
- H.323: complete, vertically integrated suite of protocols for multimedia conferencing: signaling, registration, admission control, transport, codecs
- SIP: single component. Works with RTP, but does not mandate it. Can be combined with other protocols, services

- H.323 comes from the ITU (telephony)
- SIP comes from IETF: borrows much of its concepts from HTTP
  - SIP has Web flavor; H.323 has telephony flavor
- SIP uses KISS principle: **K**eep **I**t **S**imple **S**tupid

Outline

- Multimedia networking applications
- Voice over IP
  - Skype
- Protocols for real-time conversational applications
  - RTP
  - SIP
- **Network support for multimedia**
  - Can the network (instead of application) provide mechanisms to support multimedia content delivery

Network Support for Multimedia

Approach	Granularity	Guarantee	Mechanisms	Complex	Deployed?
Making best of best effort service	All traffic treated equally	None or soft	No network support (all at application)	low	everywhere
Differentiated service	Traffic "class"	None of soft	Packet market, scheduling, policing.	med	some
Per-connection QoS	Per-connection flow	Soft or hard after flow admitted	Packet market, scheduling, policing, call admission	high	little to none

Differentiated services

- Want “qualitative” service classes
  - “behaves like a wire”
  - relative service distinction: Platinum, Gold, Silver
- **scalability:** simple functions in network core, relatively complex functions at edge routers (or hosts)
  - signaling, maintaining per-flow router state difficult with large number of flows
- Don’t define define service classes, provide functional components to build service classes

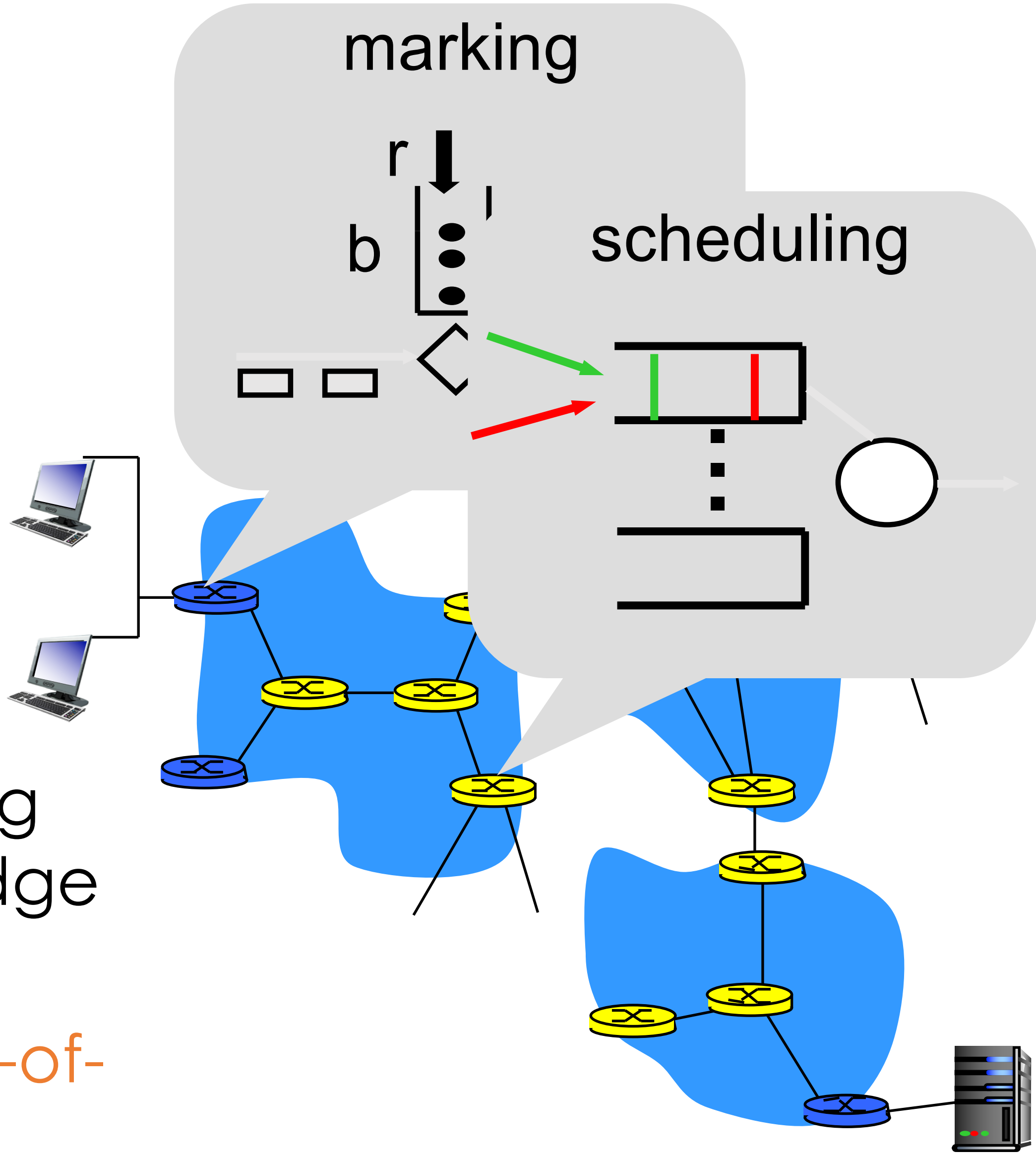
Diffserv Architecture

edge router:

- per-flow traffic management
- marks packets as in-profile and out-profile

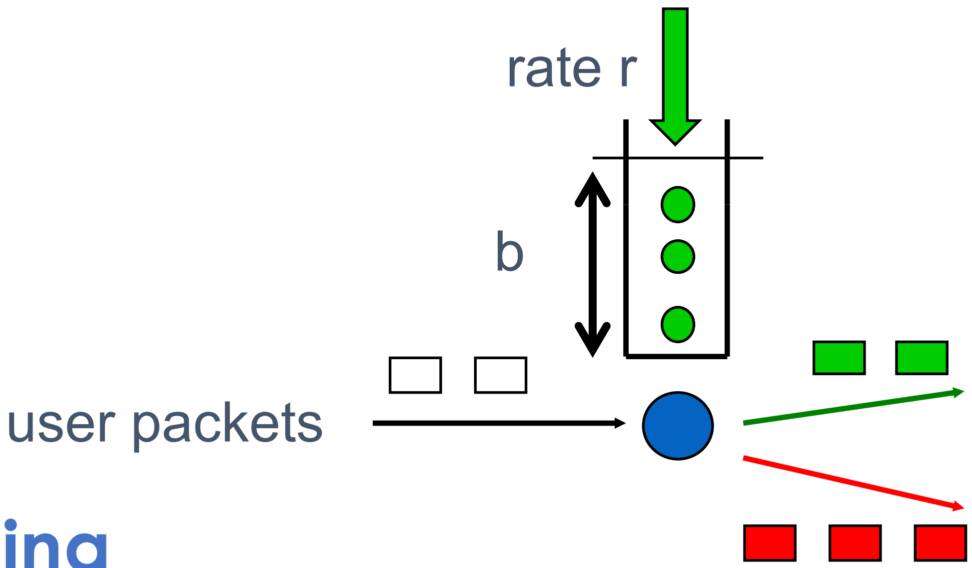
core router:

- per class traffic management
- buffering and scheduling based on marking at edge
- preference given to in-profile packets over out-of-profile packets



Edge-Router Packet Marking

- **Profile:** pre-negotiated rate r, bucket size b
- Packet marking at edge based on per-flow profile



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

Diffserv Packet Marking: Details

- 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)
  - determine PHB that the packet will receive
  - 2 bits currently unused