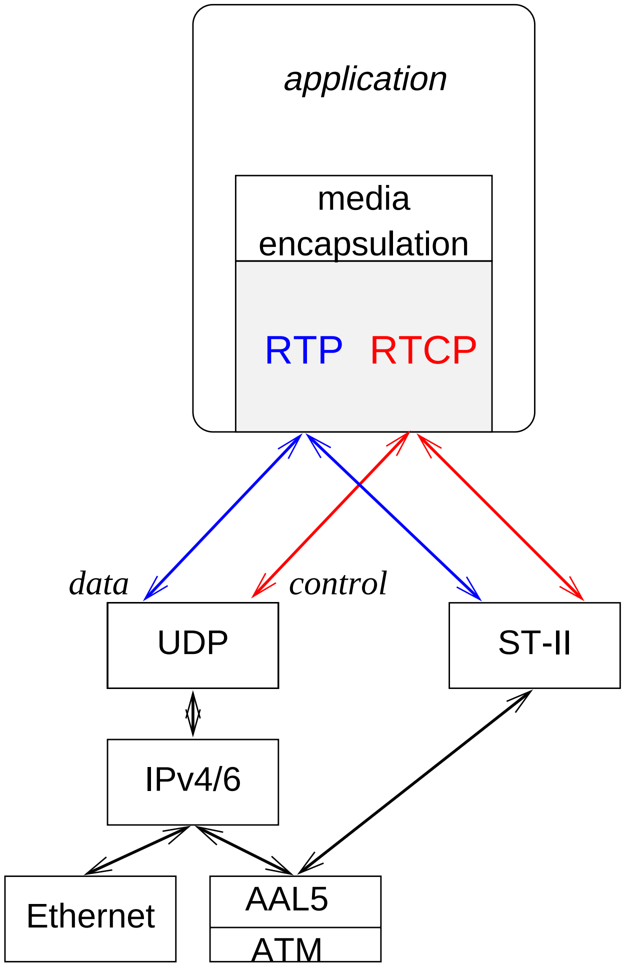
Real-Time Transport Protocol (RTP)

RTP

- protocol goals
- mixers and translators
- control: awareness, QOS feedback
- media adaptation

RTP – the big picture



RTP = Real-time transport protocol

- only part of puzzle: reservations, OS, ...
- product of Internet Engineering Task Force, AVT WG
- RFC 1889, 1890 (to be revised)
- initiated by ITU H.323 (conferencing, Internet telephony), RTSP, SIP, ...
- support for functions, but does not restrict implementation
- compression for low-bandwidth networks: CRTP (RFC 2508)

RTP goals

lightweight: specification and implementation

flexible: provide mechanism, don't dictate algorithms

protocol-neutral: UDP/IP, ST-II, IPX, ATM-AALx, …

scalable: unicast, multicast from 2 to $O(10^7)$

separate control/data: some functions may be taken over by conference control protocol

secure: support for encryption, possibly authentication

Data transport – RTP

Real-Time Transport Protocol (RTP) = data + control

data: timing, loss detection, content labeling, talkspurts, encryption

control: (RTCP) ➠ periodic with $T \sim$ population

- QOS feedback
- membership estimation
- loop detection

RTP functions

- segmentation/reassembly done by UDP (or similar)
- resequencing (if needed)
- loss detection for quality estimation, recovery
- intra-media synchronization: remove delay jitter through playout buffer
- intra-media synchronization: drifting sampling clocks
- inter-media synchronization (lip sync between audio and video)
- quality-of-service feedback and rate adaptation
- source identification

RTP mixers, translators, . . .

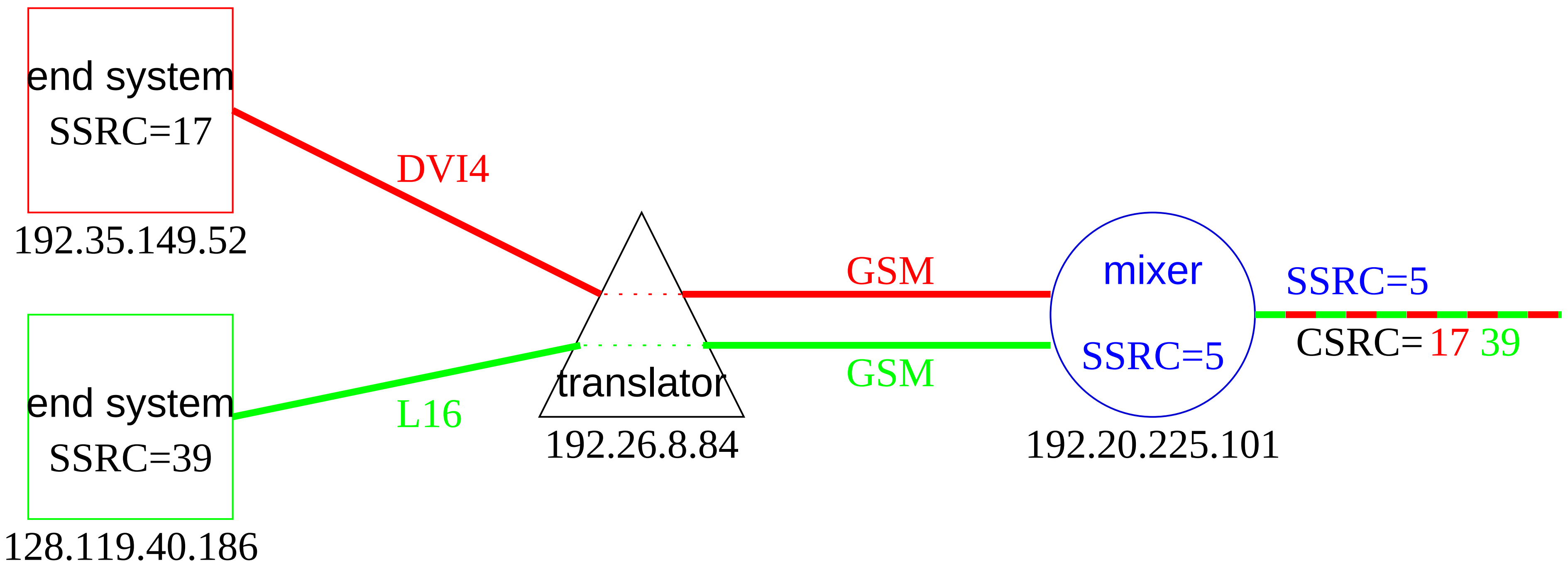
mixer:

- several media stream ➠ one new stream (new encoding)
- mixer: reduced bandwidth networks (dial-up)
- appears as new source, with own identifier

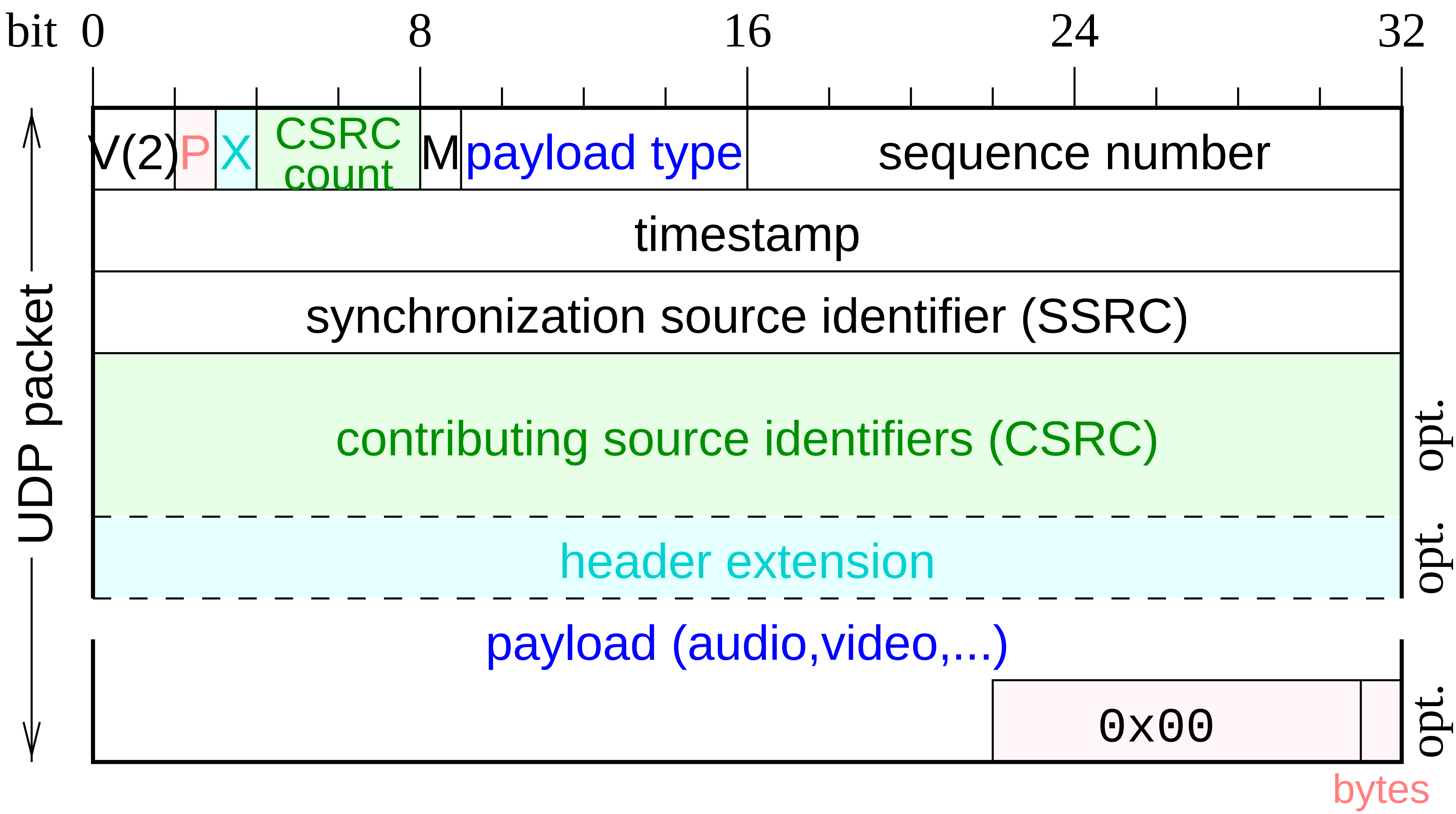
translator:

- single media stream
- *may* convert encoding
- protocol translation (native ATM $\leftrightarrow$ IP), firewall
- all packets: source address = translator address

RTP mixers, translators, . . .



RTP packet header



RTP packet header

Payload type: audio/video encoding method; may change during session

SSRC: sychronization source ➠ sources pick at random
➠ may change after *collision*!

sequence number: +1 each packet ➠ gaps ≡ loss

P: padding (for encryption) ➠ last byte has padding count

M: marker bit; frame, start of talkspurt ➠ delay adjustment

CC: content source count (for mixers)

CSRC: identifiers of those contributing to (mixed into) packet

RTP timestamp

- +1 per sample (e.g., 160 for 20 ms packets @ 8000 Hz)
- random starting value
- different fixed rate for each audio PT
- 90 kHz for video
- several video frames may have same timestamp
- ➠ gaps ≡ silence
- time per packet may vary
- split video frame (carefully. . .) across packets
- typical: 20 to 100 ms of audio

RTP in a network

- typical: UDP, no fixed port; RTCP port = RTP port (even) + 1
- typical UDP size limited to few hundred bytes (OS, network, fragmentation)
- native ATM: directly into AAL5 frame
- encapsulation (length field) for others
- typically: one media (audio, video, . . .) per port pair
- exception: bundled MPEG

RTP control protocol – types

stackable packets, similar to data packets

sender report (SR): bytes send ➠ estimate rate;
timestamp ➠ synchronization

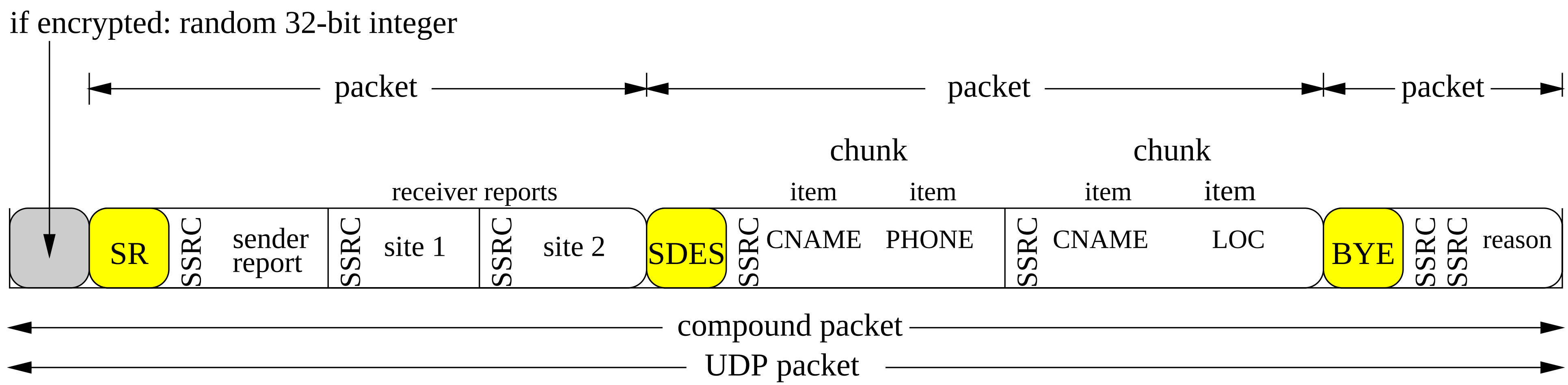
reception reports (RR): number of packets sent and expected ➠ loss, interarrival jitter, round-trip delay

source description (SDES): name, email, location, . . .
CNAME (canonical name = user@host) identifies user across media

explicit leave (BYE): in addition to time-out

extensions (APP): application-specific (none yet)

RTCP packet structure



RTCP announcement interval computation

Goals:

- estimate current # & identities of participants – dynamic
- source description ("SDES") ➠ who's talking?
- quality-of-service feedback ➠ adjust sender rate
- to $O(1000)$ participants, few % of data

➠ randomized response with rate $\downarrow$ as members $\uparrow$

- group size limited by tolerable age of status
- gives active senders more bandwidth
- soft state: delete if silent

RTCP bandwidth scaling

- every participant: periodically multicast RTCP packet to same group as data
- ➠ everybody knows (eventually) who's out there
- session bandwidth:
  - single audio stream
  - $\sum$ of concurrently active video streams

RTCP bandwidth scaling

- sender period T:

$$T = \frac{\text{\# of senders}}{0.25 \cdot 0.05 \cdot \text{session bw}} \cdot \text{avg. RTCP packet size}$$

- receivers:

$$T = \frac{\text{\# of receivers}}{0.75 \cdot 0.05 \cdot \text{session bw}} \cdot \text{avg. RTCP packet size}$$

- next packet = last packet + max(5 s, T) · random(0.5 . . . 1.5)
- randomization prevents “bunching”
- to reduce RTCP bandwidth, alternate between SDES components

RTCP sender reports (SR)

SSRC of sender: identifies source of data

NTP timestamp: when report was sent

RTP timestamp: corresponding “RTP time” ➠ lip sync

sender’s packet count: total number sent

sender’s octet count: total number sent

followed by zero or more receiver report

RTCP receiver reports (RR)

SSRC of source: identifies who's being reported on

fraction lost: binary fraction

cumulative number of packets lost: long-term loss

highest sequence number received: compare losses, disconnect

interarrival jitter: smoothed interpacket distortion

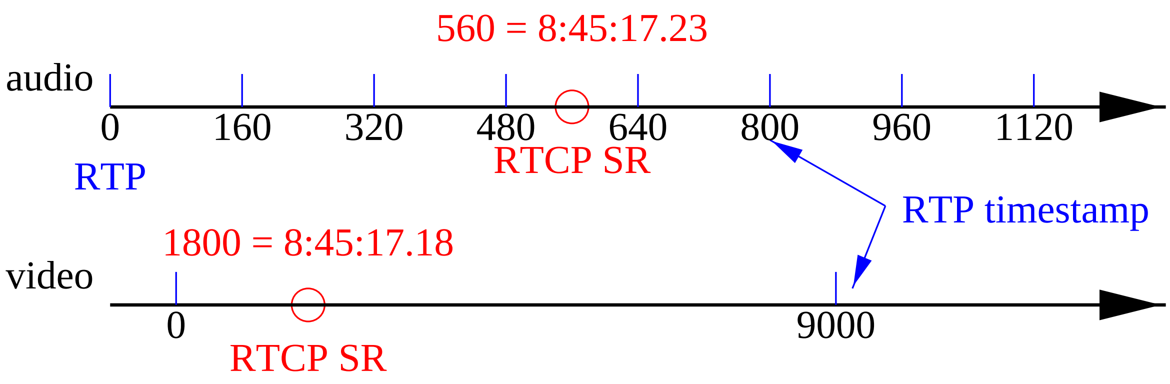
LSR: time last SR heard

DLSR: delay since last SR

Intermedia synchronization

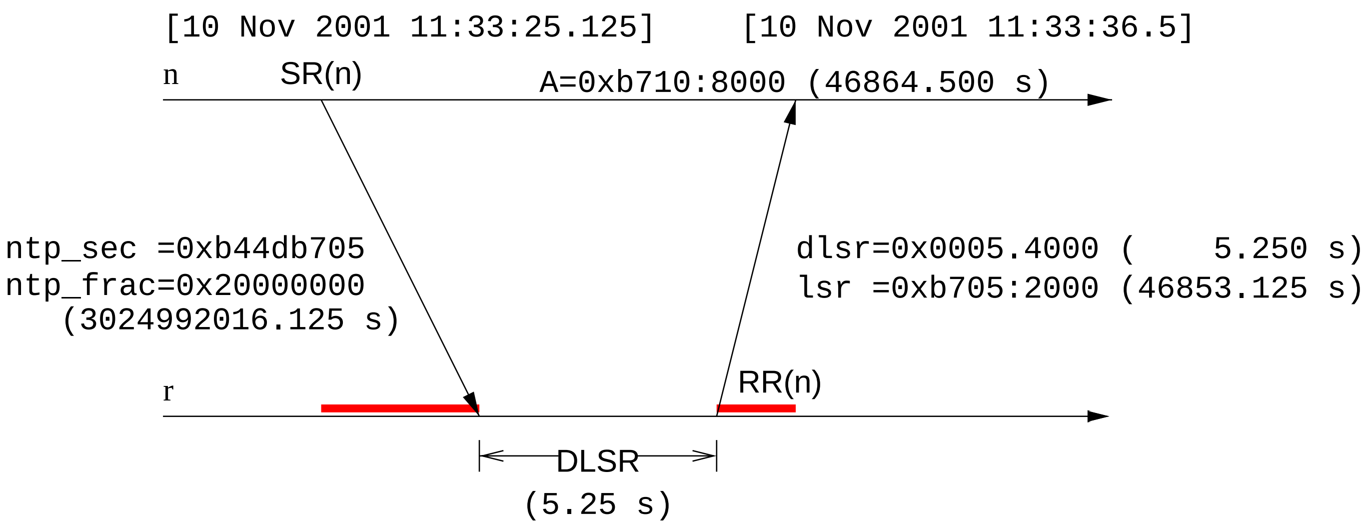
= sync different streams (audio, video, slides, . . .)

- timestamps are offset with random intervals
- may not tick at nominal rate
- SRs correlate "real" time (wallclock time) with RTP ts



Round-trip delay estimation

compute round-trip delay between data sender and receiver




```
A      0xb710:8000 (46864.500 s)
DLSR -0x0005:4000 (    5.250 s)
LSR  -0xb705:2000 (46853.125 s)
--------------------------------
delay 0x   6:2000 (    6.125 s)
```

RTP: Large groups

How do manage large groups?

- “movie at ten”
- channel surfing

➠ reconsideration: pause and recompute interval

- conditional reconsideration: only if group size estimate increases
- unconditional reconsideration: always
- reverse reconsideration to avoid time-outs

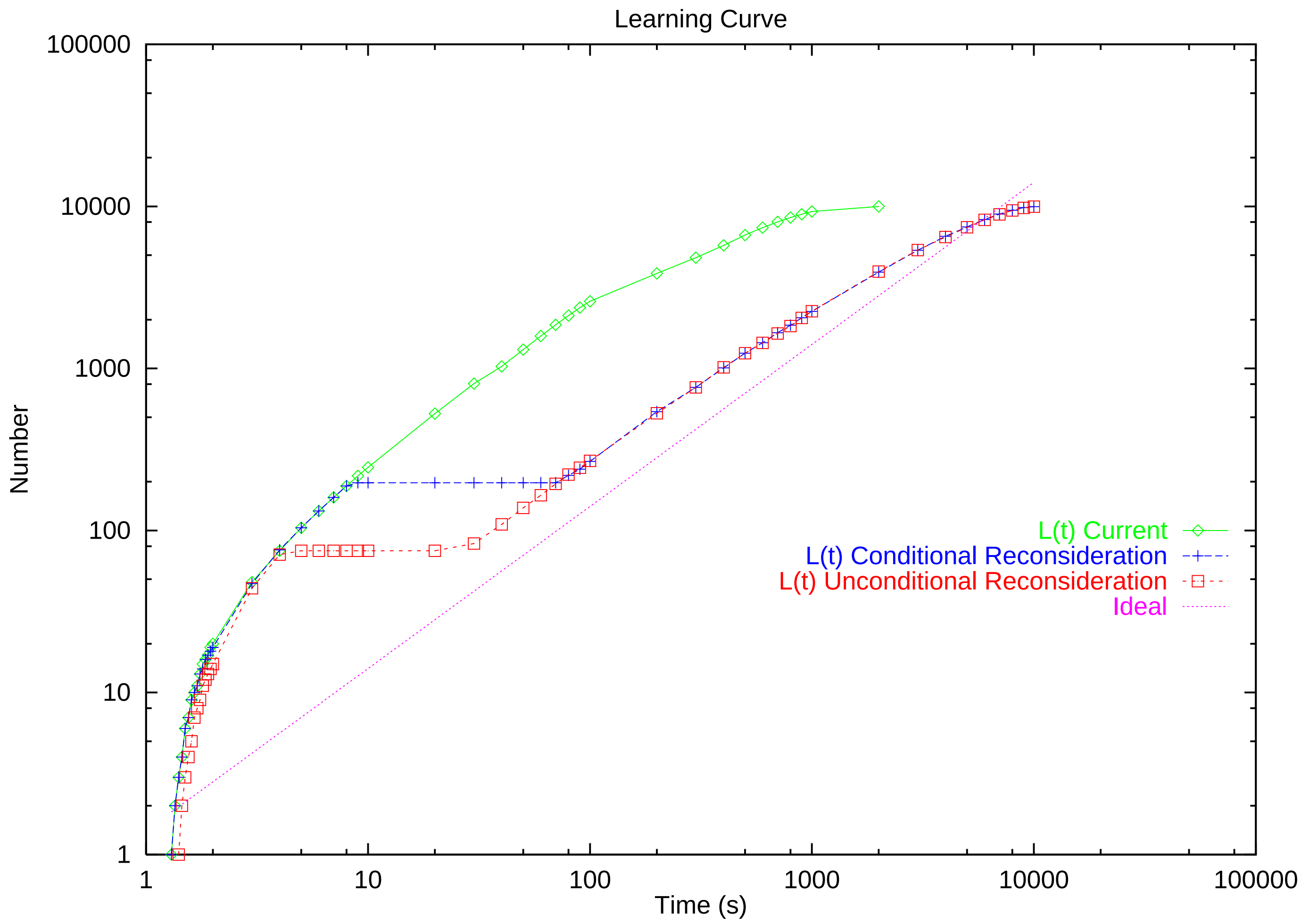
BYE floods

- avoid BYE floods: don’t send BYE if no RTCP
- reconsideration

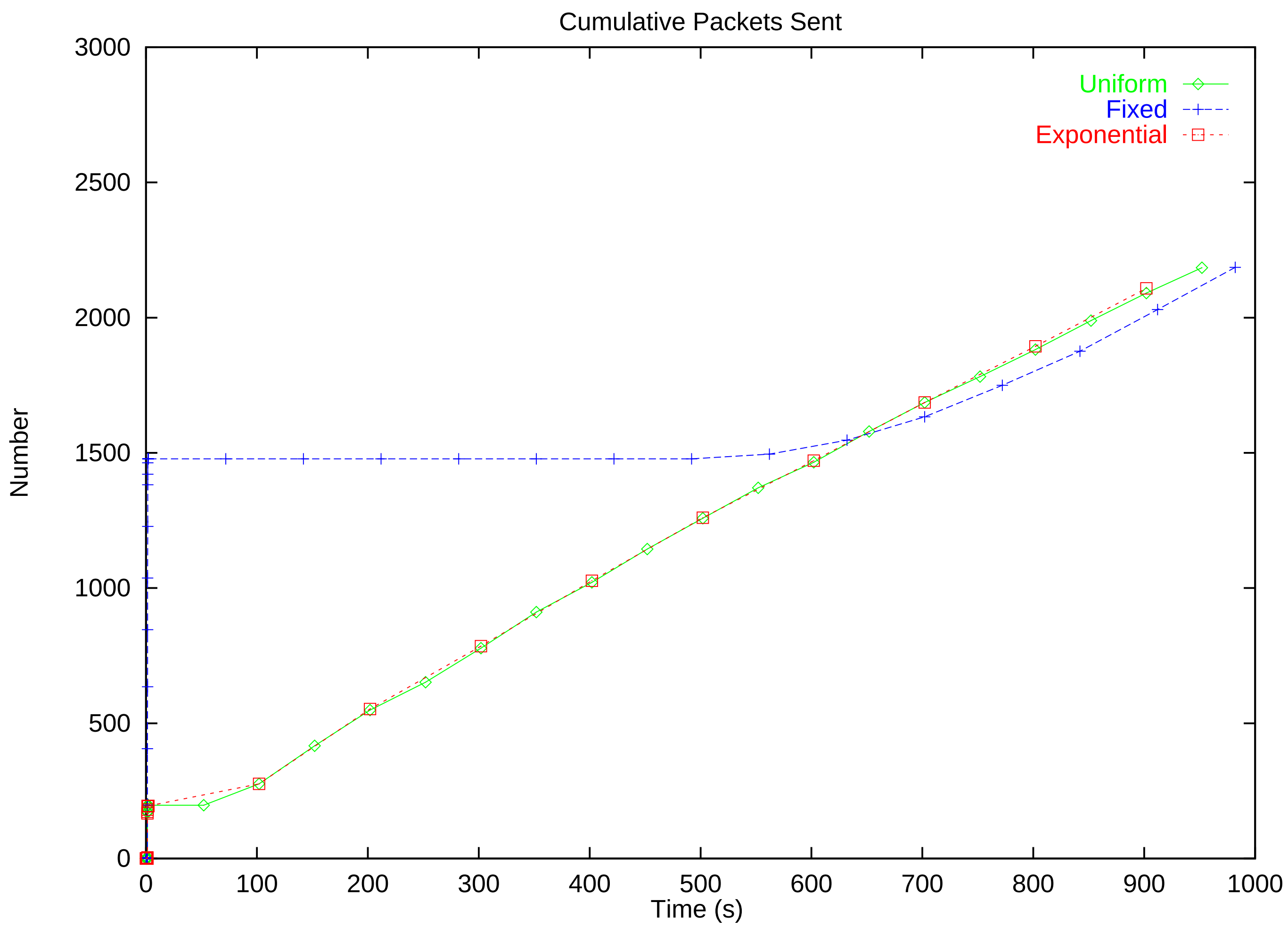
More general:

- general bandwidth sharing problem
- “squeaky wheel” network management

Reconsideration: learning curve



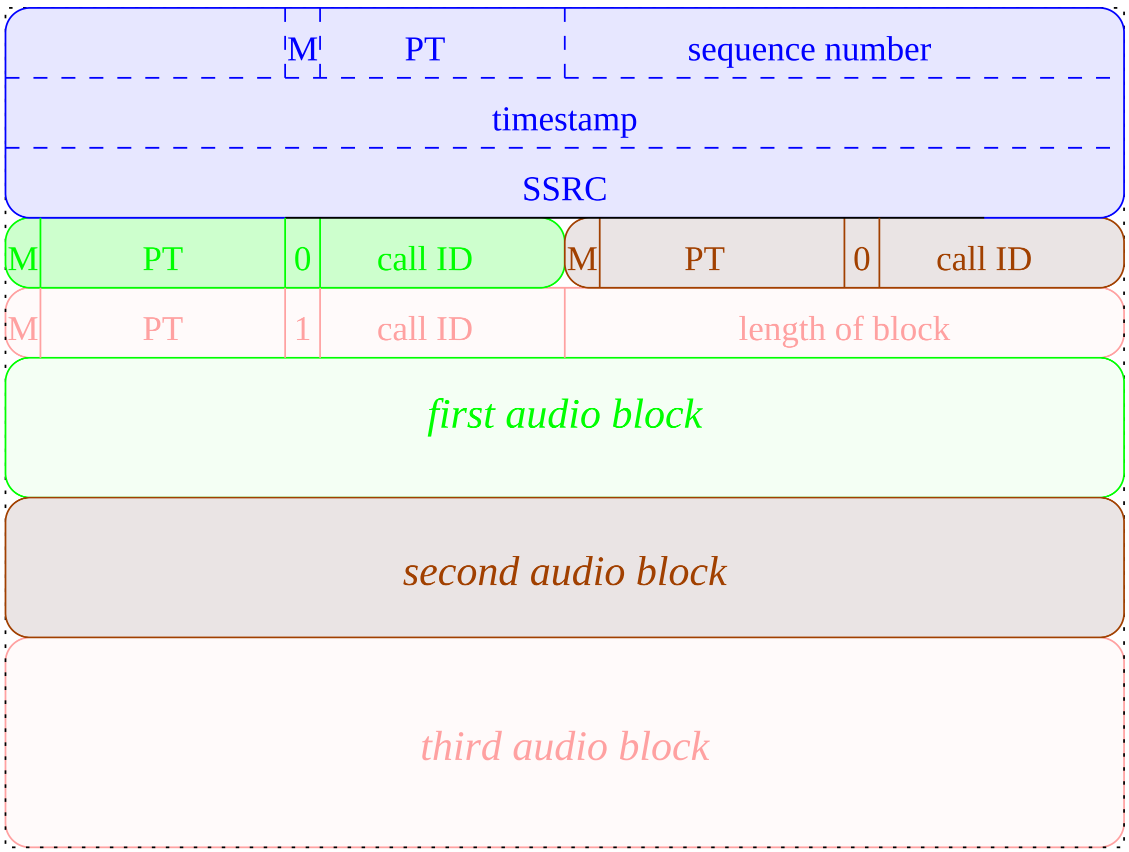
Reconsideration: influence of delay



RTP: Aggregation

- interconnected IPTel gateways ➠ several RTP streams to same destination
- high overhead: G.729, 30 ms packetization ➠ 30 bytes audio, 40 bytes IP + UDP + RTP headers
- with ATM: efficiency = 28%
- solution: bundle several calls into single RTP session

RTP: Aggregation



- for 24 channels ➠ efficiency ↑ 89%
- signal call-ID using SIP

Collision detection and resolution

Collision:

- two sources may pick the same SSRC (“birthday problem”)
- probability: about 10^{-4} if 1000 session members join more or less simultaneously
- but: don’t pick one you know about already ➠ probability much lower unless everyone joins at the same time
- send BYE for old, pick a new identifier

Loop detection

- forward packet to same multicast group (directly or through translators)
- looks similar to collision, but changing SSRC doesn't help
- look at RTCP packets

RTP for the masses

- for 14.4 kb/s stream: 90 B/s $\approx$ 1 new site/s
- takes $\approx$ 3 hours to get to know 10,000 people ➠
  - who cares? (Nielsen!)
  - useless for QOS feedback
  - control rate too high
- ➠ statistical sample (sender determines rate): send value $[0, 1]$; pick random value; if $<$, lucky winner ➠ needs to be adaptive
- ➠ report just to sender, instead of multicast

Adaptive applications

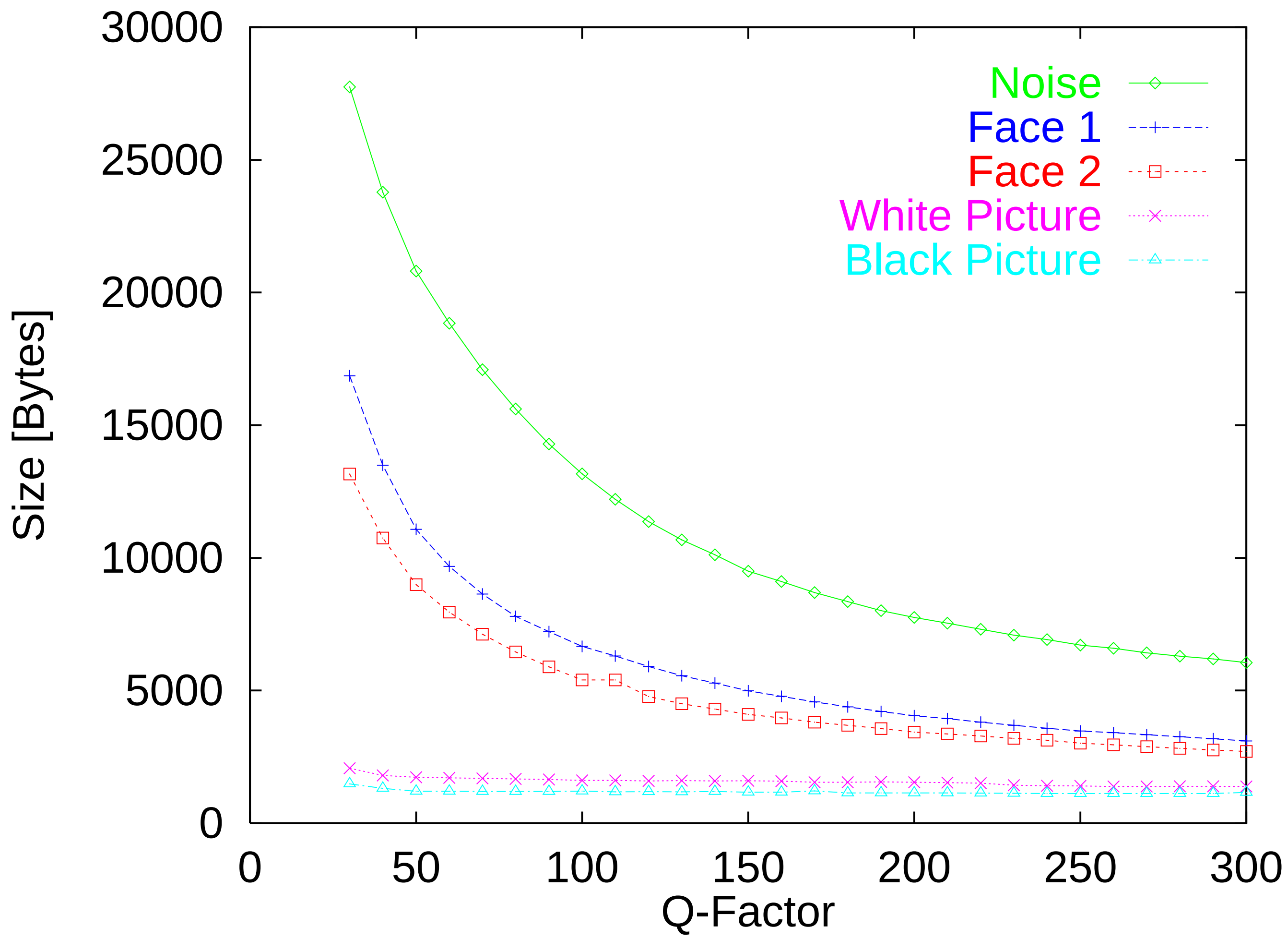
Adaptive applications

Multimedia applications can adjust their data rates:
Audio: encoding parameters (MPEG L3), encoding, sampling rate, mono/stereo

encoding	sampling rate	bit rate
LPC	8,000	5,600
GSM	8,000	13,200
DVI4	8,000	32,000
μ-law	8,000	64,000
DVI4	16,000	64,000
a range of DVI4 and MPEG L3		
L16 stereo	44,100	1,411,200

Adaptive applications

Video: frame rate, quantization, image resolution, encoding



Application control

- networks with QoS guarantees:
  - QoS at call set-up, guaranteed
  - long call durations ➠ network load may change
  - “wrong” guess ➠ rejected calls or low quality
- networks w/o QoS or shared reserved link:
  - adapt application to available bandwidth
  - share bandwidth fairly with TCP?
  - lowest common demoninator ➠ mixers, translators

TCP-friendly applications

- avoid race due to FEC, aggressive retransmission
- push aside TCP applications (sometimes ok. . .)
- avoid congestion collapse
- avoid being but in “penalty box”
- time scale?

TCP-friendly adaptation

- rate computation (e.g.,):
  - use additive-increase, multiplicative-decrease
  - use loss/RTT equation: throughput = $\frac{1.22}{R\sqrt{p}}$, where R is the round-trip time and $p \approx$ loss fraction
- mechanisms:
  - TCP ACKs, without retransmission $\longrightarrow$ overhead, no multicast
  - RTCP RR $\longrightarrow$ delay, metric?

RTP: Status and Issues

Compression: differential compression for low-speed point-to-point links ➠ compress IP, UDP, RTP into 1–2 bytes

Aggregation: trunking of packet streams or Internet telephony gateways

Large groups: RTCP feedback for O(10,000); sampling

RTP (RFC 1889, RFC 1890) $\longrightarrow$ draft standard

RTP Header Compression

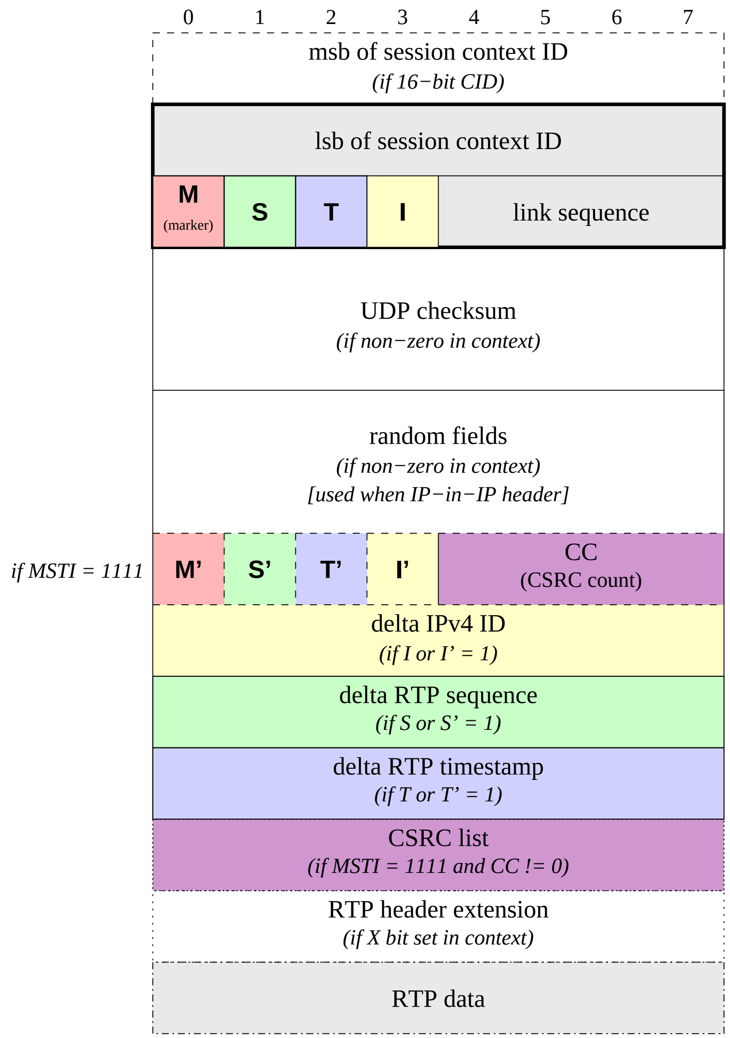
- large overhead for IP + UDP + RTP headers: 40 bytes
- CRTP = lossless differential compression that reduces overhead to two bytes on (low-speed) point-to-point links
- derived from VJ TCP/IP header compression
- context: IP address, port, RTP SSRC
- IP: only packet ID changes
- UDP: only checksum
- RTP: second-order difference of timestamp and sequence number is zero
- resynchronization by NAK $\longrightarrow$ not good for high BER, delay

RTP Header Compression

- link layer indicates `FULL_HEADER`, `COMPRESSED_UDP`, `COMPRESSED_RTP`, `CONTEXT_STATE` (no IP header)
- differences are encoded as variable-length fields:

-16384	C0 00 00
-129	C0 3F 7F
-128	80 00
-1	80 7F
0	00
127	7F
128	80 80
16383	BF FF
16384	C0 40 00
4194303	FF FF FF

CRTP Packet Header



Some RTP Implementations

tool	who	media	RSVP	adaptive
NeVoT	GMD Fokus	audio	yes	not yet
vic	LBNL	video	no	no
vat	LBNL	audio	no	no
rat	UCL	audio	no	no
Rendezvous	INRIA	A/V	no	yes
NetMeeting	Microsoft	A/V	no	no
IP/TV	Cisco	A/V	no	no
RM G2	Real	A/V	no	yes

http://www.cs.columbia.edu/~hgs/rtp/