L17 – Network Performance and CDNs

Jit Biswas

Cohort 1: TT7&8 (1.409-10)

Cohort 2: TT24&25 (2.503-4)

Note: Some slides are from slide sets of Kurose and Ross Ed 6 Ch 1 and Ch 7

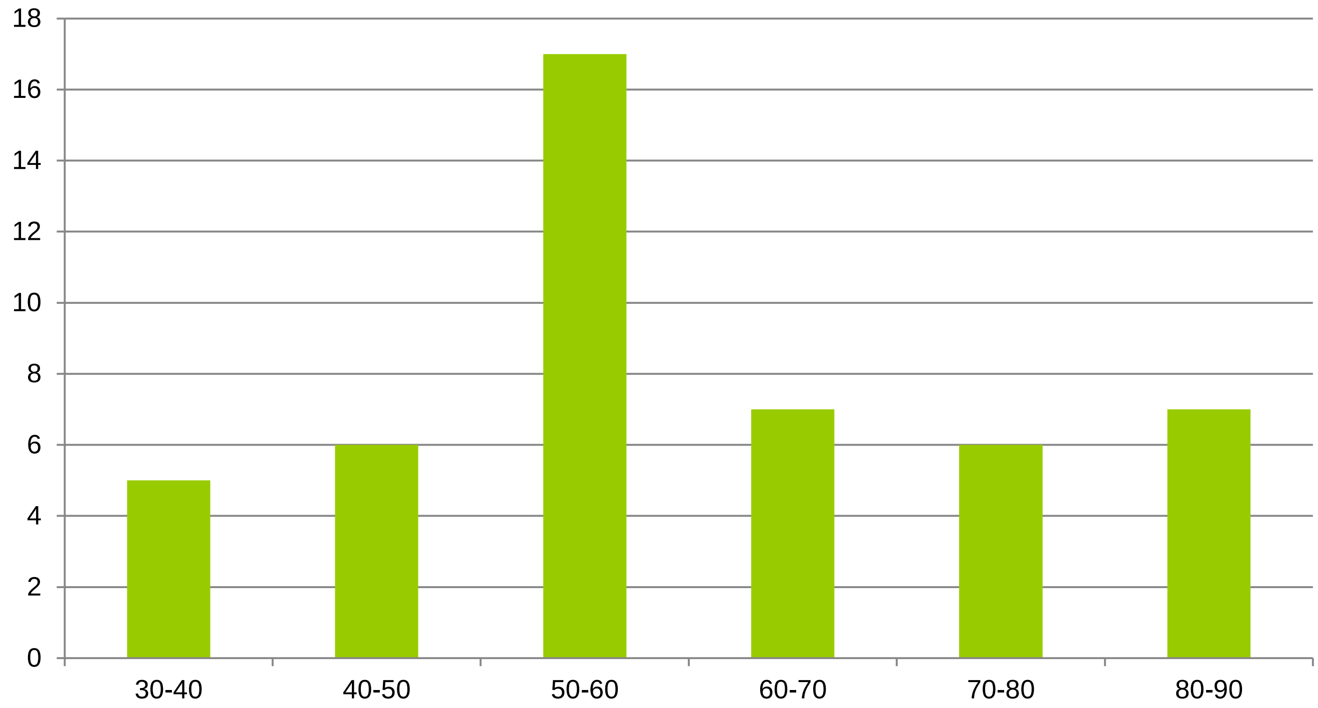
Introduction

- This lecture:
  - Estimating Network Performance
  - Bottlenecks
  - Content Distribution Networks
- Working towards learning objectives
  - Design optimized network topology for given problem settings
  - Judge and evaluate a provided network setup

Midterm Grades

- Available on eDimension
- They will be displayed out of hundred
  - Will be converted to 22% of final grade
  - Lab, project and activity grades compensate for low exam grades
- If you would like to see your graded exam, please contact your TA
  - C1 – Yee Ching
  - C2 – John
- You can see them during their office hours this week

Midterm Grades

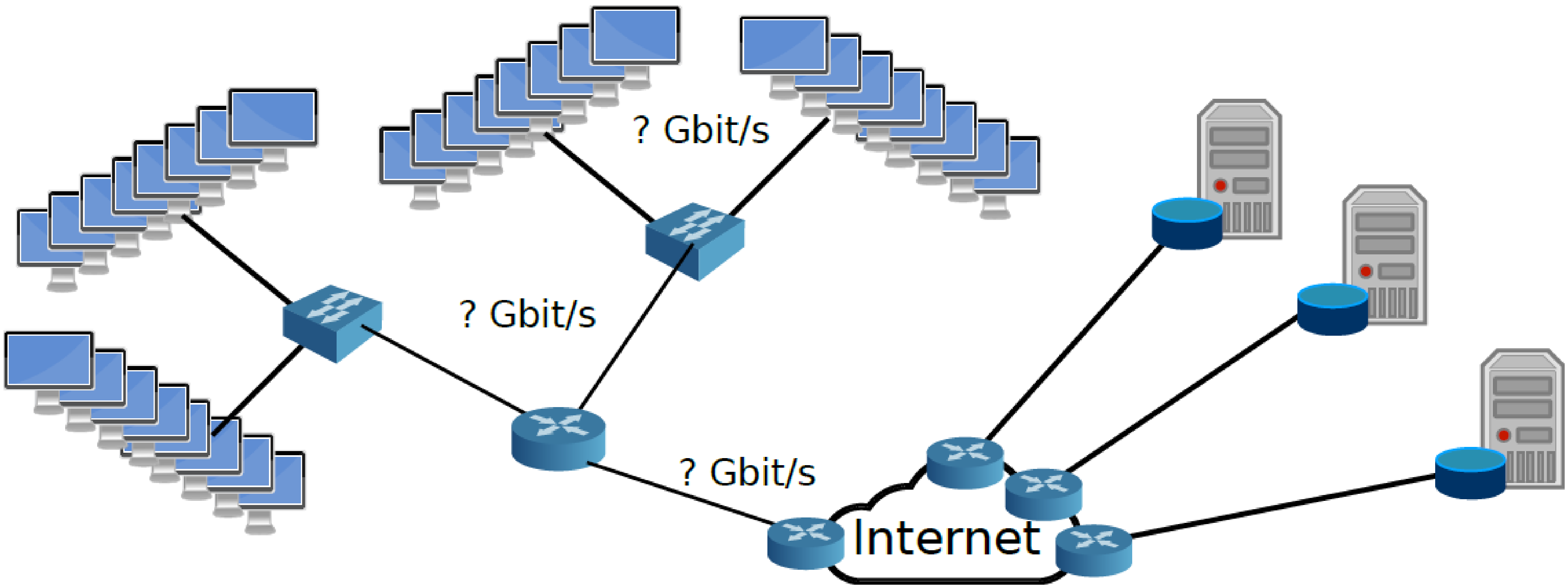


- Min: 34.5
- Max: 88.5
- Mean: 59.5
- Median: 58
- Standard deviation: 14.8

Performance Networking

Performance Networking

- So far, we have discussed network services
- How to build a high performance network?
  - Locally, and Internet Scale?



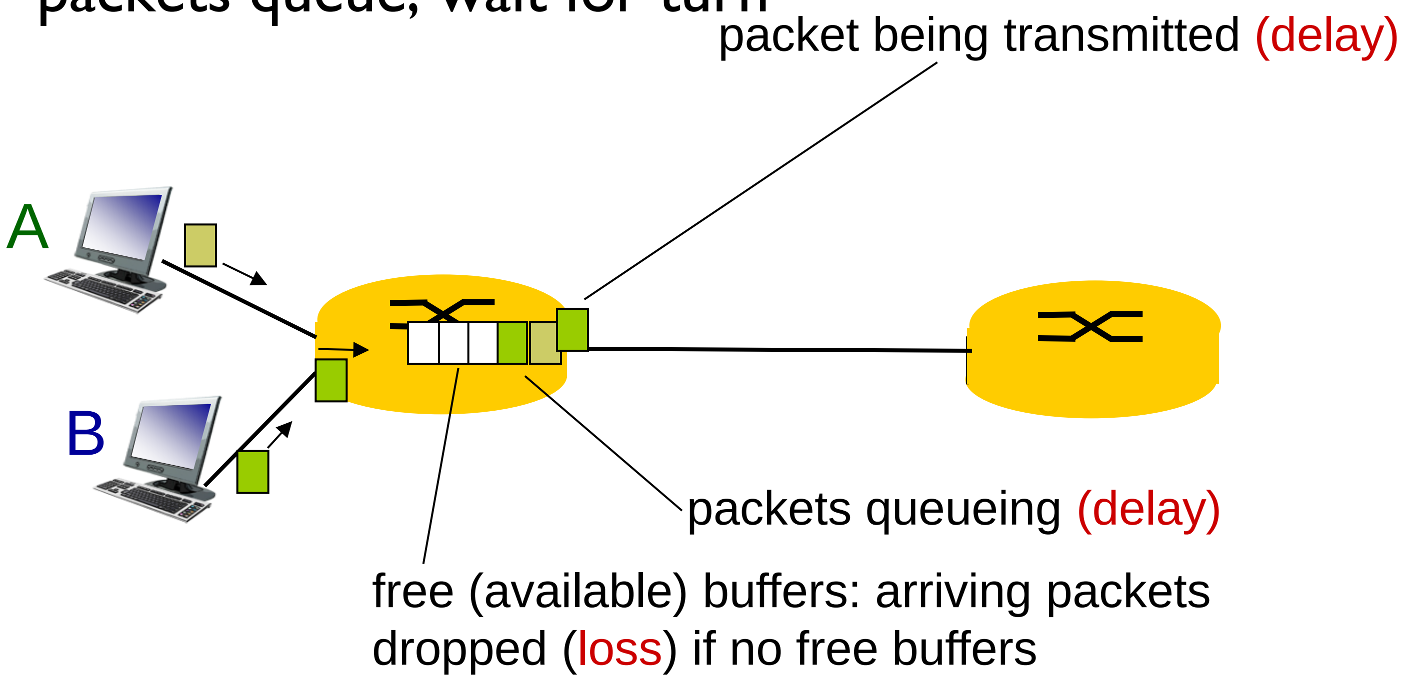
Review: Delay types

- Propagation delay
  - Speed of light -> Distance traveled
  - Ideally, communication partner is close
- Bandwidth/ transmission delay
  - How many bits per second?
  - Which link is bottleneck?
- Competing traffic/ queuing delay
  - How much delay spent in buffering queues?
  - Up/Download?

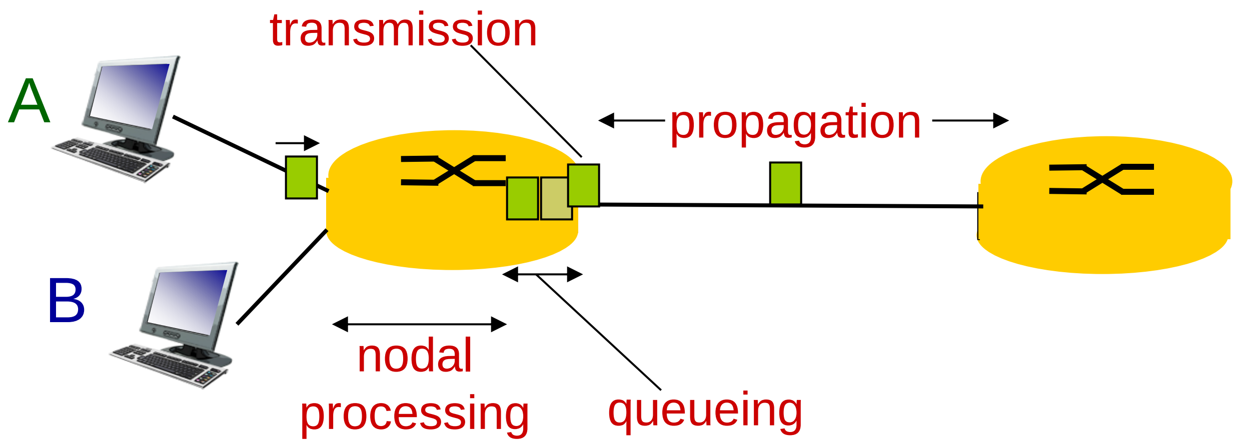
What happens when packet arrives at router?

packets *queue* in router buffers

- packet arrival rate to link (temporarily) exceeds output link capacity
- packets queue, wait for turn



Four sources of packet delay



Single "hop" nodal delay (i.e., delay from one node to *immediate* next one):

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{proc}: nodal processing

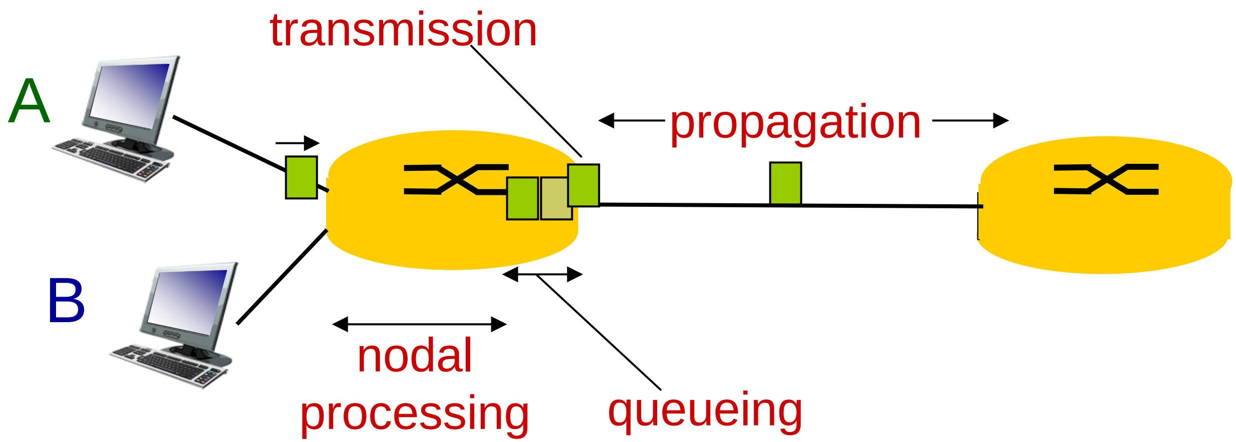
- check for bit errors (by checksum in packet header)
- determine output link (by destination IP address in packet header)
- typically < msec

d_{queue}: queueing delay

- waiting time for packet to get to front of the queue for the output link
- depends on congestion level of router (i.e., how much *other* users are also sending data)

Source: Chapter 1 slide set, J.F Kurose and K.W. Ross, Ed 6

Four sources of packet delay



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{trans}: transmission delay:

- *L*: packet length (bits)
- *R*: link *bandwidth (bps)*
- $d_{trans} = L/R$

d_{prop}: propagation delay:

- *d*: length of physical link
- *s*: propagation speed in medium (~2x10^8 m/sec – 2/3 speed of light in vacuum)
- $d_{prop} = d/s$

d_{trans} and d_{prop} *very* different

1. Transmission delay: time to push whole packet (all the bits) from router to (beginning of) link – how quickly we can do this depends on the link technology, specifically its bandwidth (e.g., Ethernet has 10 Mbps bandwidth)
2. Propagation delay: time for packet to move from beginning to end of the link

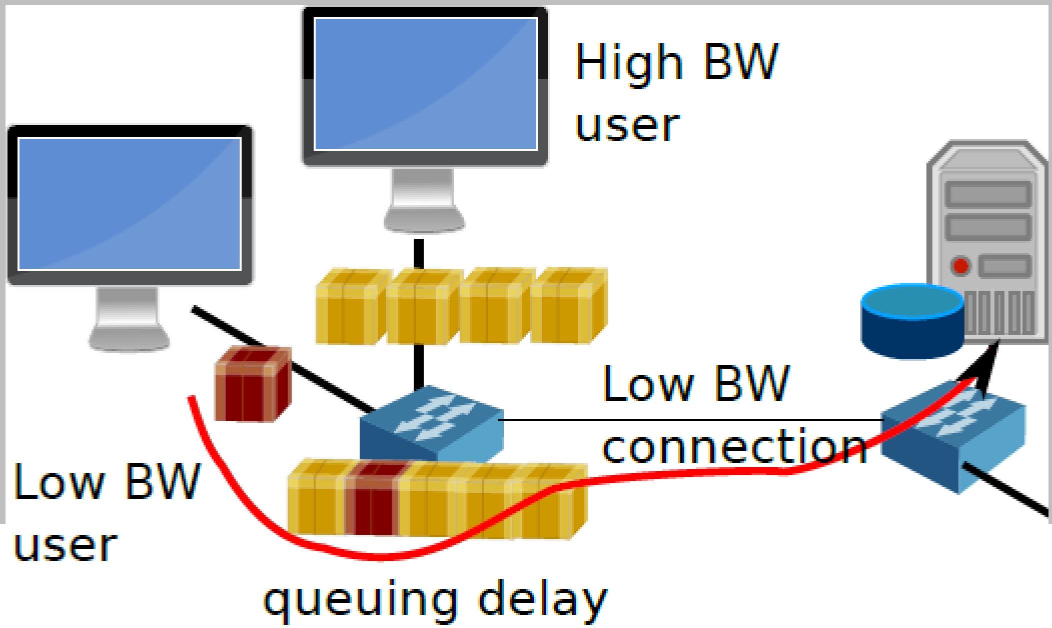
Congestion and Latency

- Links become overloaded if more traffic is scheduled to be sent than possible -> Average throughput rate will go down
- Even for low bandwidth applications, latency will also become bad. Why?

Congestion and Latency

- Links become overloaded if more traffic is scheduled to be sent than possible -> Average throughput rate will go down
- Even for low bandwidth applications, latency will also become bad. Why?

- Incoming packets will queue up in buffers
- The waiting time in queue will increase latency



Estimate Traffic Generation

- How much traffic are you generating?
  - When browsing Facebook
  - When listening to spotify or soundcloud
  - When watching 480p (640x480px, 30fps) video
  - When watching FullHD 1080p (1920x1080px, 60fps) video

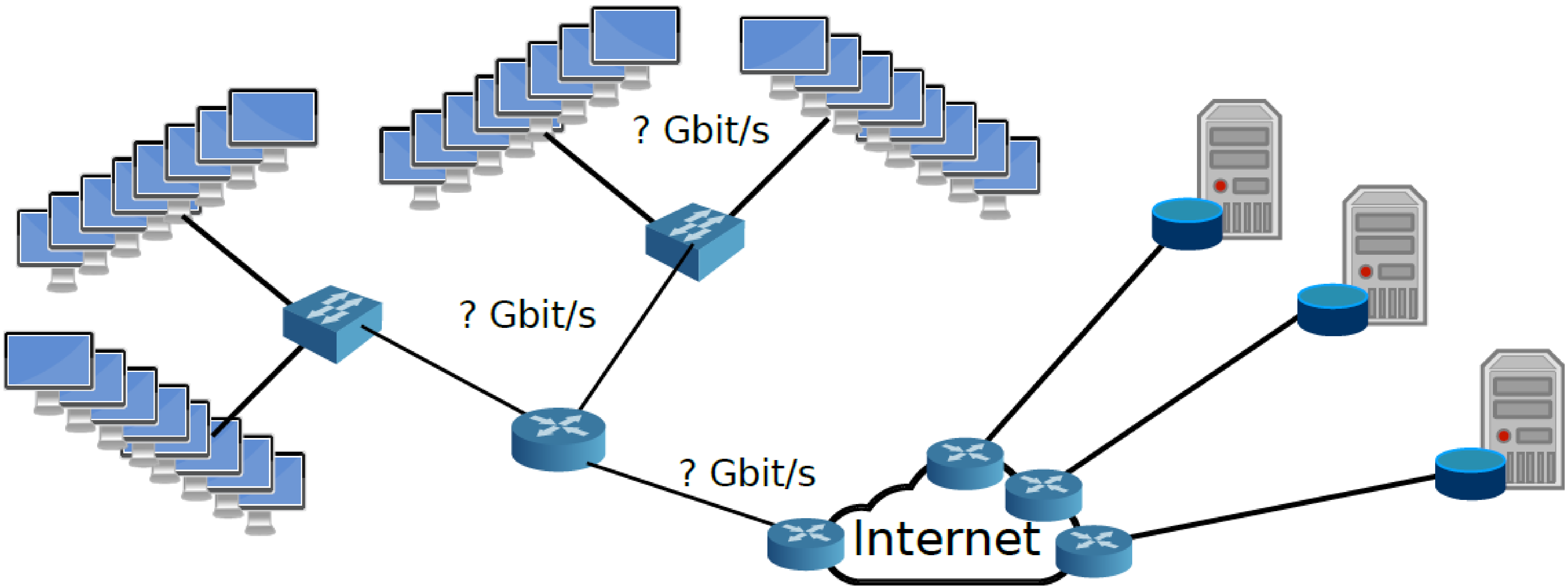
Estimate Traffic Generation

- How much traffic are you generating?
  - When browsing Facebook
  - When listening to spotify or soundcloud
  - When watching 480p (640x480px, 30fps) video
  - When watching FullHD 1080p (1920x1080px, 60fps) video

- Estimates:
  - Facebook 50 kb/s
  - Music 250 kb/s
  - 480p 1000 kb/s
  - 1080p 5000 kb/s

Sizing links part 1

- How should we size links in the following network, assuming users like you?



Recap Congestion Probability

- It is easy to compute congestion probability
  - If uniform distribution of user load is assumed
- Example: 100 Mb/s link, shared by 35 users
- Each user requires 10Mb/s when active
  - *On average*, users are only active 10% of the time
- What is the chance that the links is overloaded?

Recap Congestion Probability

- It is easy to compute congestion probability
  - If uniform distribution of user load is assumed
- Example: 100 Mb/s link, shared by 35 users
- Each user requires 10Mb/s when active
  - *On average*, users are only active 10% of the time
- What is the chance that the links is overloaded?

- Binomial distribution yields probability of overload
- For error threshold of 0.05%, 35 users can share link
  - Probability of exceeding bandwidth is < .0004

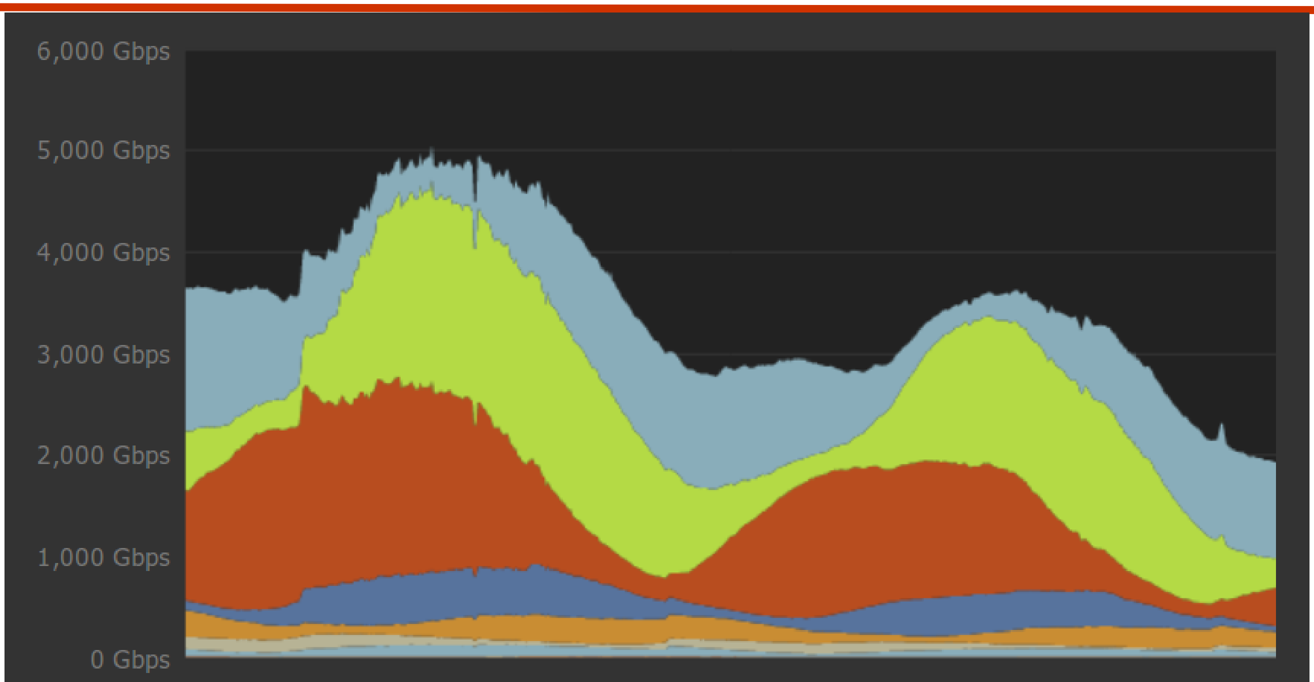
$$P(X \leq 10) = \sum_{i=0}^{10} \binom{35}{i} (0.1)^i (0.9)^{35-i} = 0.99957$$

$$P(X > 10) = 1 - P(X \leq 10) = 0.00042..$$

Bottlenecks

- End user connections are getting better and better
  - 500mbit 802.11ac, 1000mbit fiber connections
- What is SUTD's internet bandwidth? Lets assume 500 Mbit/s
- How much aggregated traffic for all of Singapore?
  - Lets assume 5M users, 10% users active, 4Mbit link
  - We require roughly 2Tbit/s connection!
- How can we reduce this load on the undersea cables?

Actual Distribution



	current	peak
Total Bandwidth Used:	**1.9 Tbps**	**5.1 Tbps**
North America	940.7 Gbps	1.4 Tbps
Europe	287.8 Gbps	2 Tbps
Asia	382.2 Gbps	2 Tbps
Russia	55.1 Gbps	502.4 Gbps
South America	155.2 Gbps	268.3 Gbps
Oceania	47.7 Gbps	141.6 Gbps
Middle East	43.9 Gbps	112.6 Gbps
Africa	5.6 Gbps	14.5 Gbps
Central America	6.2 Gbps	9.2 Gbps

Source: Steam network 20 Nov 2017

Traffic Type US

Upstream		Downstream		Aggregate	
BitTorrent	18.37%	Netflix	35.15%	Netflix	32.72%
YouTube	13.13%	YouTube	17.53%	YouTube	17.31%
Netflix	10.33%	Amazon Video	4.26%	HTTP - OTHER	4.14%
SSL - OTHER	8.55%	HTTP - OTHER	4.19%	Amazon Video	3.96%
Google Cloud	6.98%	iTunes	2.91%	SSL - OTHER	3.12%
iCloud	5.98%	Hulu	2.68%	BitTorrent	2.85%
HTTP - OTHER	3.70%	SSL - OTHER	2.53%	iTunes	2.67%
Facebook	3.04%	Xbox One Games Download	2.18%	Hulu	2.47%
FaceTime	2.50%	Facebook	1.89%	Xbox One Games Download	2.15%
Skype	1.75%	BitTorrent	1.73%	Facebook	2.01%
	69.32%		74.33%		72.72%

sandvine

Source: Sandvine, data for North America 2016

Traffic Type Asia

Rank	Upstream	2016	Downstream		Aggregate	Share
1	Facebook	14.85%	YouTube	20.87%	YouTube	19.16%
2	SSL - OTHER	14.02%	Facebook	13.97%	Facebook	14.07%
3	Google Cloud	9.28%	HTTP - OTHER	9.36%	HTTP - OTHER	9.32%
4	HTTP - OTHER	8.92%	SSL - OTHER	6.85%	SSL - OTHER	7.62%
5	YouTube	5.01%	Instagram	6.66%	Instagram	6.31%
6	Snapchat	4.36%	Snapchat	5.17%	Snapchat	5.09%
7	Instagram	3.35%	Netflix	3.72%	Google Cloud	3.56%
8	BitTorrent	2.16%	iTunes	3.02%	Netflix	3.41%
9	FaceTime	1.97%	Google Cloud	2.87%	iTunes	2.86%
10	iCloud	1.82%	MPEG - OTHER	2.37%	MPEG - OTHER	2.17%
		65.76%		74.87%		73.57%

sandvine

Source: Sandvine, data for Asia 2016

Increasing Network Performance

- Main load is coming from videos
- Solutions exist at different layers
  - Compression (mp3, MPEG, . . .), but no major improvements expected
  - Content distribution networks
  - Multicast of streaming traffic
- Many other schemes
  - HTTP proxies, Quality-of-Service, . . .

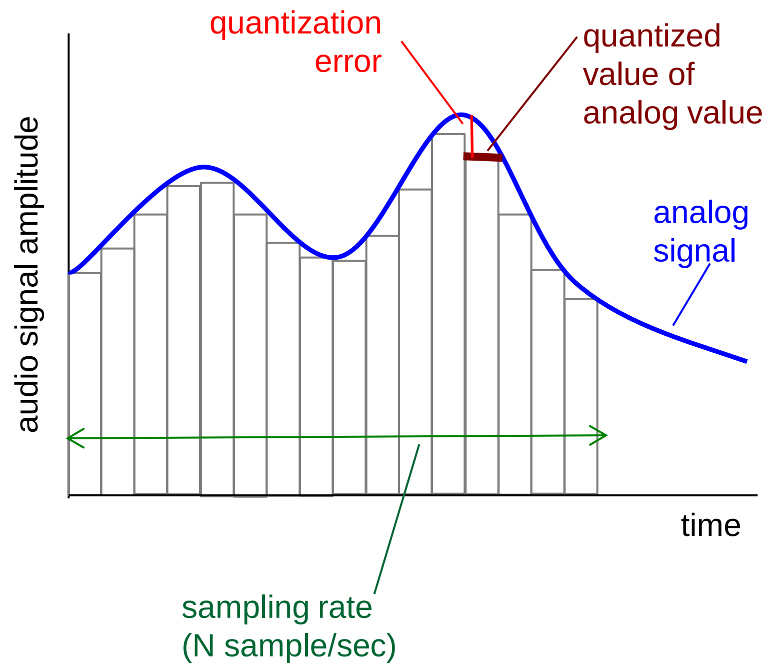
Video Streaming Basics

- Videos can have high bandwidth
  - 720p 1.5-4Mbit/s - (1280 X 720 pixels ~ 0.92 MP per frame) – HD ready
  - 1080p 3-6Mbit/s - (1920 X 1080 pixels ~ 2.07 MP per frame) – full HD
  - 4k might be around 16Mbit/s (ultra HD)
- We don’t need the data again after seeing it
  - And complete video is huge (100 GB for 4K video)
- Simple video streaming uses *sliding window* approach
  - Video data is transmitted while video is playing
  - Sender estimates required consumption rate by client
- More modern schemes allow client to fetch on demand

Multimedia: audio

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

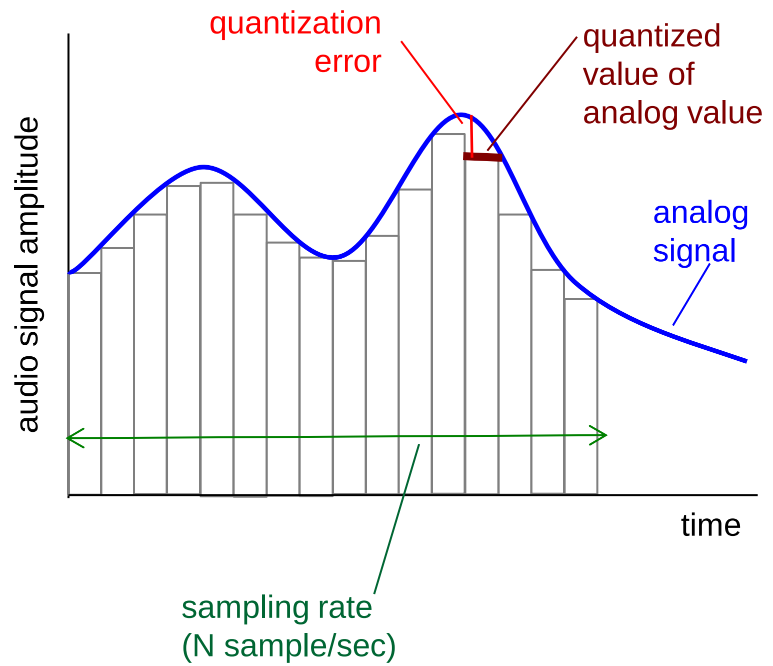
- analog audio signal sampled at constant rate
  - telephone: 8,000 samples/sec
  - CD music: 44,100 samples/sec
- each sample quantized, i.e., rounded
  - e.g., 2^8=256 possible quantized values
  - each quantized value represented by bits, e.g., 8 bits for 256 values



Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Multimedia: audio

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



example rates

- CD: 1.411 Mbps
- MP3: 96, 128, 160 kbps
- Internet telephony: 5.3 kbps and up

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Multimedia: video

- video: sequence of images displayed at constant rate
  - e.g. 24 images/sec
- digital image: array of pixels
  - each pixel represented by bits
- coding: use redundancy *within* and *between* images to decrease # bits used to encode image
  - spatial (within image)
  - temporal (from one image to next)

spatial coding example: instead of sending N values of same color (all purple), send only two values: color value (purple) and number of repeated values (N)

frame i

temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame i+1

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

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)

spatial coding example: instead of sending N values of same color (all purple), send only two values: color value (purple) and number of repeated values (N)

frame i

temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame i+1

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Multimedia networking: 3 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)
  - e.g., YouTube, Netflix, Hulu
- *conversational* voice/video over IP
  - interactive nature of human-to-human conversation limits delay tolerance
  - e.g., Skype
- *streaming live* audio, video
  - e.g., live sporting event (futbol)

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Types of streaming

- *UDP Streaming*
  - *Server transmits at a rate that matches client's consumption rate*
  - Requires RTP (Real-Time Transport Protocol)
  - Requires separate out of band control thru RTSP
- *HTTP Streaming*
  - TCP based, with congestion control – can cause delay
  - Can go beyond firewall because of HTTP traffic
  - Majority of streaming systems use this method

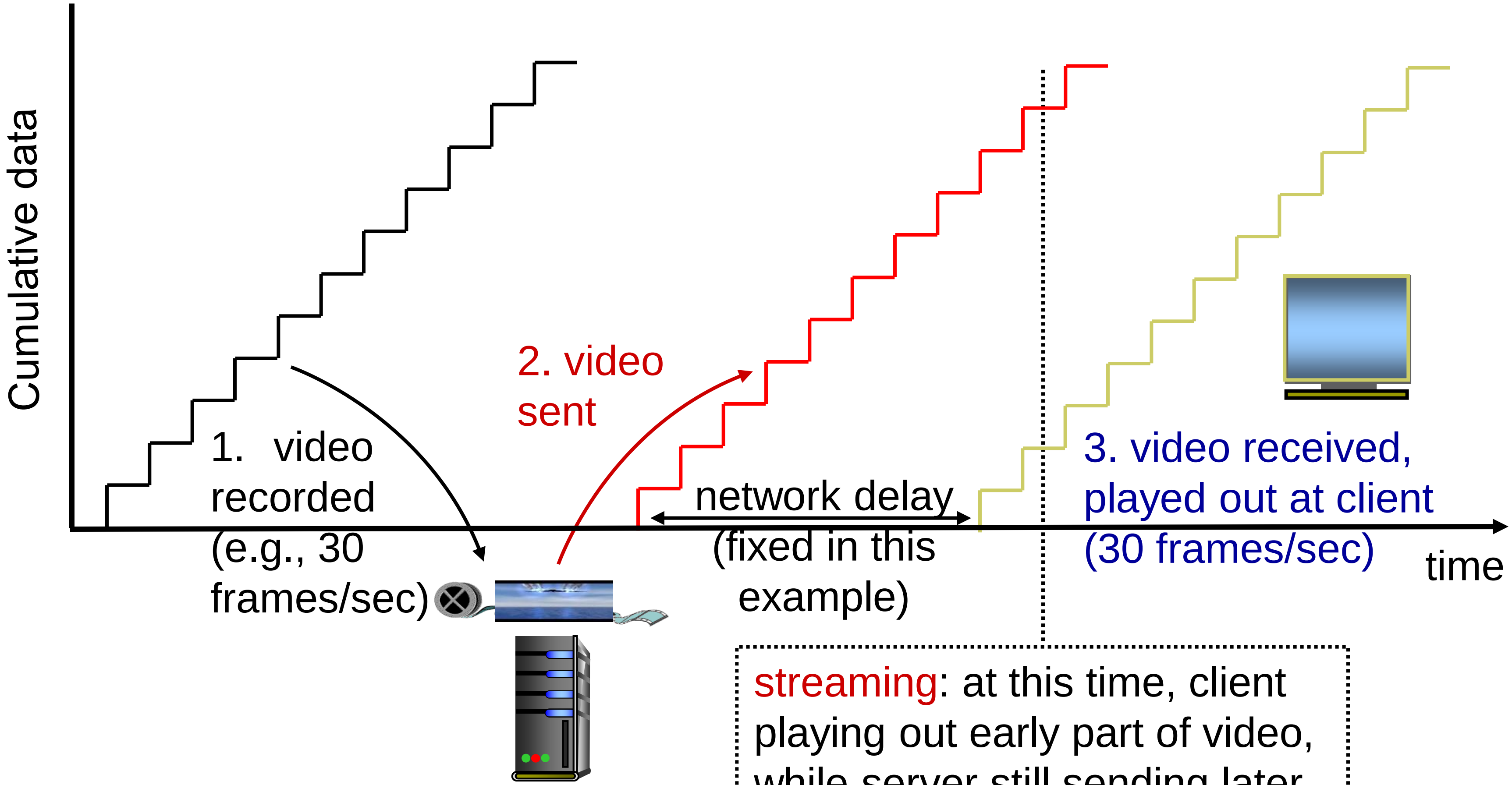
  Eg. Youtube, Netflix …
- *Adaptive HTTP streaming*

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Drawbacks of UDP Streaming

- *Fails to provide continuous playout*
  - *Unpredictable and variable bandwidth between server and client*
  - Poor user experience
- *Requires separate media control server*
  - Essential to handle client-server interactivity requests
  - Increases cost and complexity
- *Often blocked by firewalls*
  - Prevents users behind firewalls to receive UDP video

Streaming stored video:



streaming: at this time, client playing out early part of video, while server still sending later part of video

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

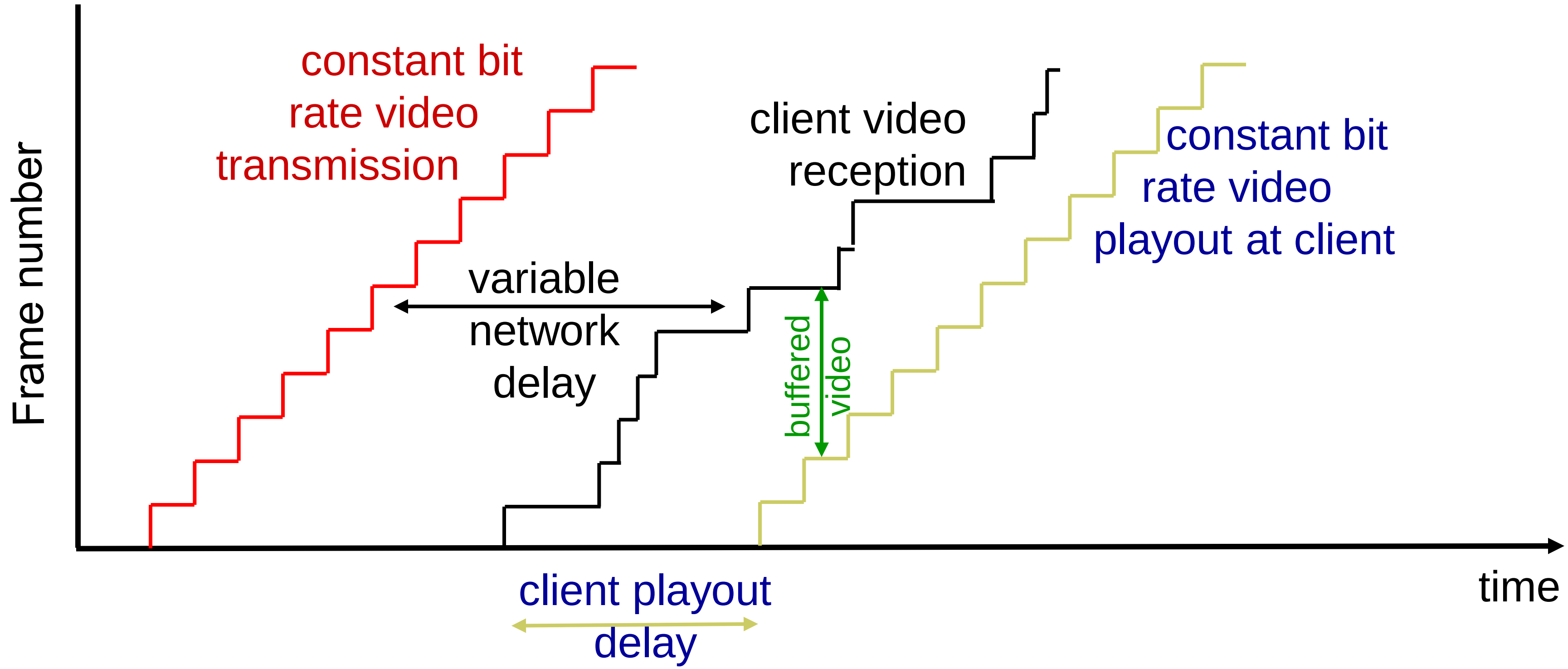
Streaming stored video: challenges

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

- continuous playout constraint: once client playout begins, playback must match original timing
  - … but network delays are variable (jitter), so will need client-side buffer to match playout requirements
- other challenges:
  - client interactivity: pause, fast-forward, rewind, jump through video
  - video packets may be lost, retransmitted

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Streaming stored video: revisted



- Server sends at steady rate (e.g. UDP)
  - At least as big as *consumption rate*
- Network add random delays
- Pre-fetching and buffering still allow continuous playback

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Streaming multimedia: DASH

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
- *server:*
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file:* provides URLs for different chunks
- *client:*
  - periodically measures server-to-client bandwidth
  - consulting manifest, requests one chunk at a time
    - chooses maximum coding rate sustainable given current bandwidth
    - can choose different coding rates at different points in time (depending on available bandwidth at time)

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Streaming multimedia: DASH

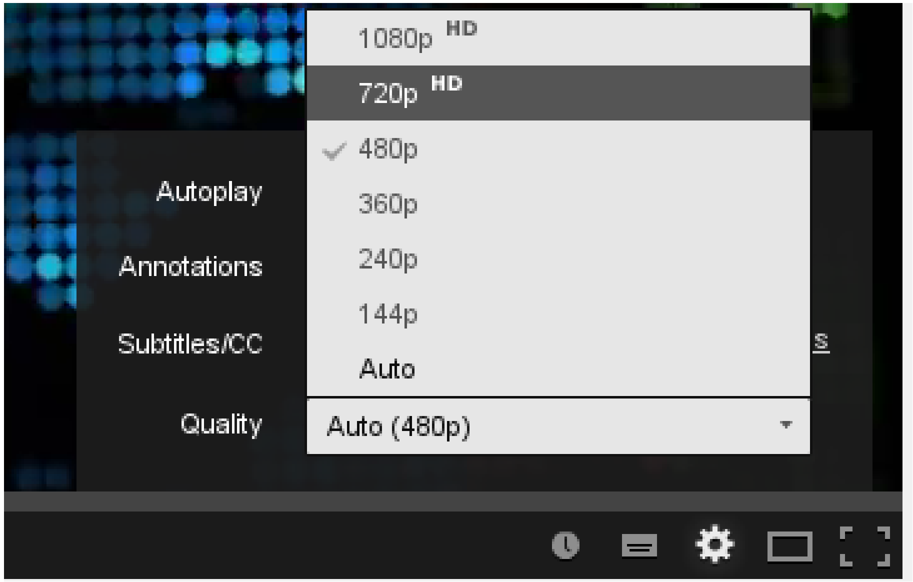
- *DASH: Dynamic, Adaptive Streaming over HTTP*
- *"intelligence"* at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request from URL server that is "close" to client or has high available bandwidth)

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Dynamic Adaptive Streaming over HTTP

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

- DASH is commonly used nowadays
- Video file is split into chunks
- Each chunk is available in different resolutions
  - *Manifest* file by server provides index
- Client will choose appropriate resolution and fetch chunks
  - TCP addresses reliable transport
  - TCP also works better with firewalls than UDP



Source: Youtube quality selector

Activity 14: Types of Multimedia Applications and Streaming

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

a) Name the three types of multimedia networking discussed in class, briefly describing each in one sentence OR a couple of keywords

b) Name the three types of streaming discussed in class, briefly describing each in one sentence OR a couple of keywords

c) Name three disadvantages of UDP Streaming

Content Delivery Networks (CDNs)

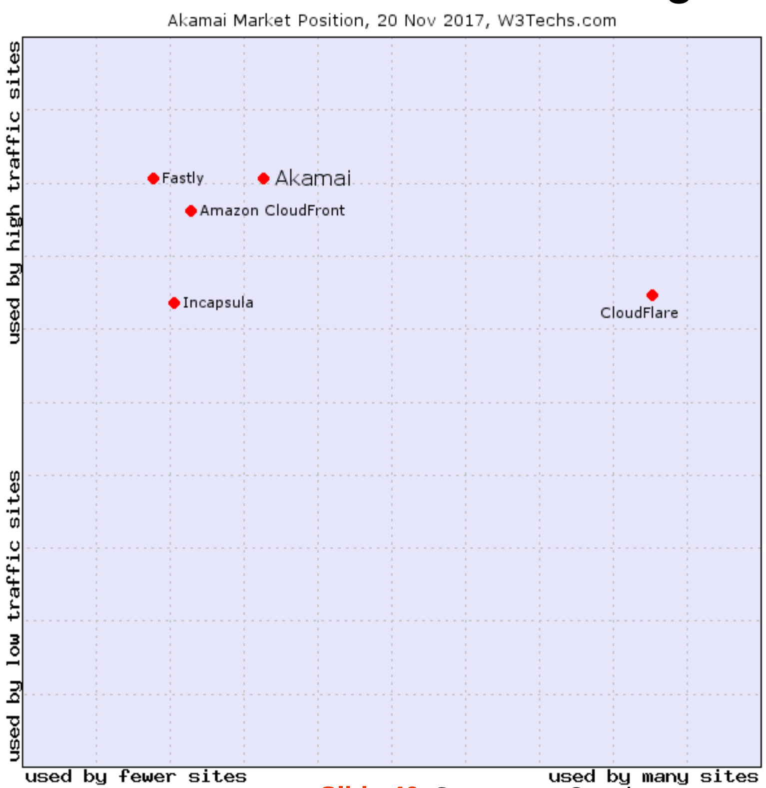
- CDNs are service providers for high traffic websites
- Simple form: *mirror* servers, replicating static content
  - Similar to mirror servers, Software companies with large downloads
  - Related to proxies we discussed, but proxies run by AS operator
- More advanced: dynamic media such as audio and video
  - Need more complicated replication of content

CDN Strategies

- Most CDNs follow one of two strategies
  - *Enter Deep:* Try to be as close to the customer as possible. Requires a large number of small clusters. Example: Akamai
  - *Bring Home:* Focus on few large clusters, connect them with high speed lines to important ISPs. Example: Limelight
- Google follows both strategies
  - Several hundred enter deep clusters with ISPs
  - About 30 bring home clusters
  - 8 mega data centers in North America and Europe
- Nowadays, many big technology companies build their own CDN
  - Amazon, Apple, Microsoft, L1 ISPs, . . .

CDN Example: Akamai

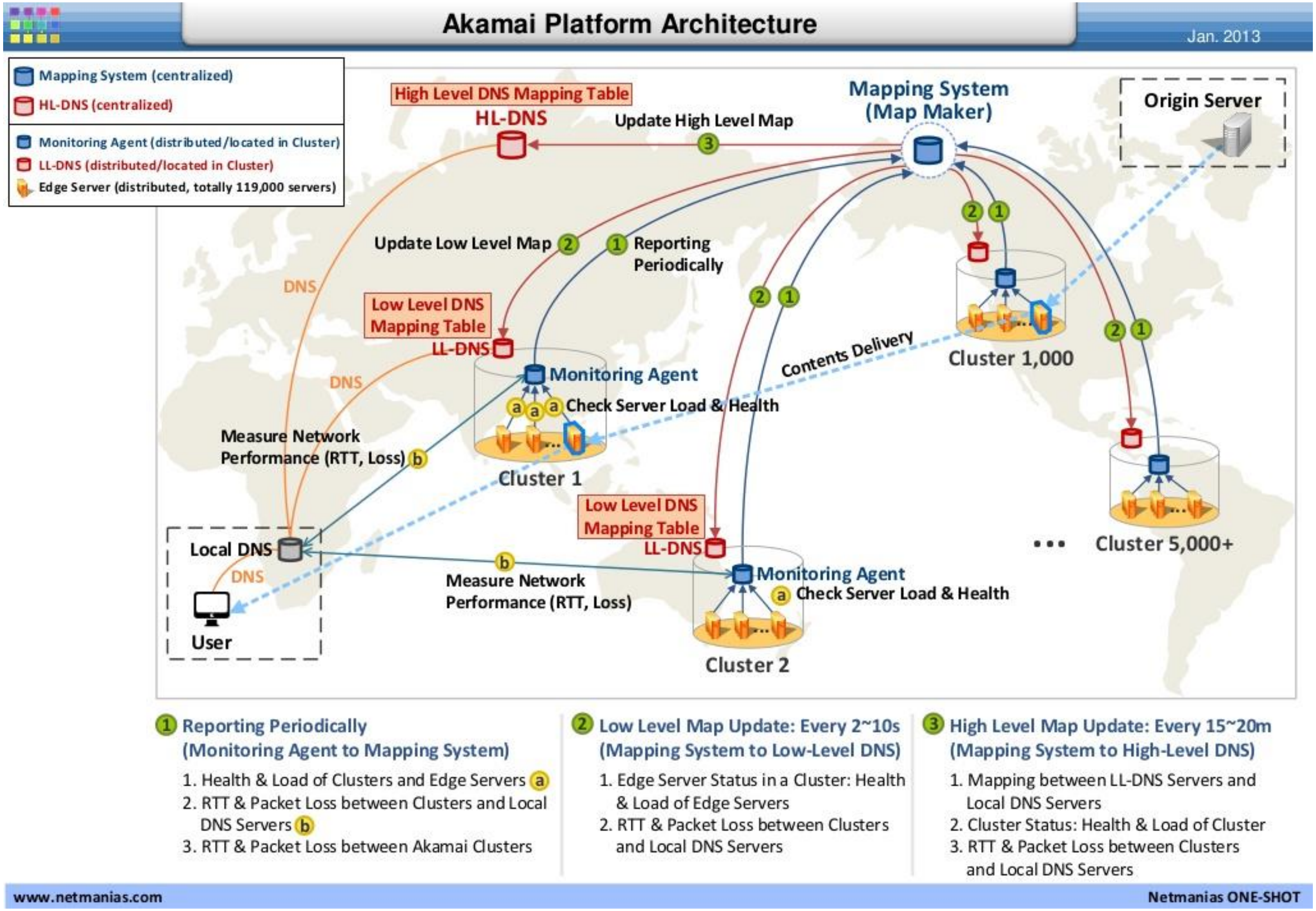
- Akamai was one of the pioneering CDNs (since before 2000)
  - They pioneered *enter deep*
- According to Akamai
  - 240,000 servers deployed in more than 130 countries at 1600 locations
  - Akamai delivers around 10% of all Web traffic
- 85% of the world's Internet users are within a single "network hop" of an Akamai server



 Source: w3techs.com

Akamai Architecture Overview

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN



Source: Netmanias (tech@netmanias.com)

CDN vs Cloud

- CDN were invented in late 90's
- Cloud and *elastic computing* was coined in the last few years
- Both want to provide resources on demand
- CDN are focused on serving static/media content
- Clouds could be anything, in particular computing and processing on demand
- DB servers can become bottleneck

Skype

- Skype provides video and audio calls
- Uses its own algorithms
- Skype's main advantage: easy to set up
  - Compared to competitors running VOIP
- Main problem: How to reach users at home?
  - NAT will prevent incoming connections to home users
- Skype solves this by requiring an active outgoing connection to regional *super nodes*
  - If two users want to talk, the respective super nodes routes traffic between users
  - Super nodes were traditionally also users, but nowadays dedicated servers are used

Voice-over-IP (VoIP)

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

- *VoIP end-end-delay requirement*: needed to maintain "conversational" aspect
  - higher delays noticeable, impair interactivity
  - < 150 msec: good
  - > 400 msec bad
  - includes application-level (packetization,playout), network delays
- *session initialization:* how does callee advertise IP address, port number, encoding algorithms?
- *value-added services:* call forwarding, screening, recording
- *emergency services:* 911

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

VoIP characteristics

- 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 of data
- application-layer header added to each chunk
- chunk+header encapsulated into UDP or TCP segment
- application sends segment into socket every 20 msec during talk spurt

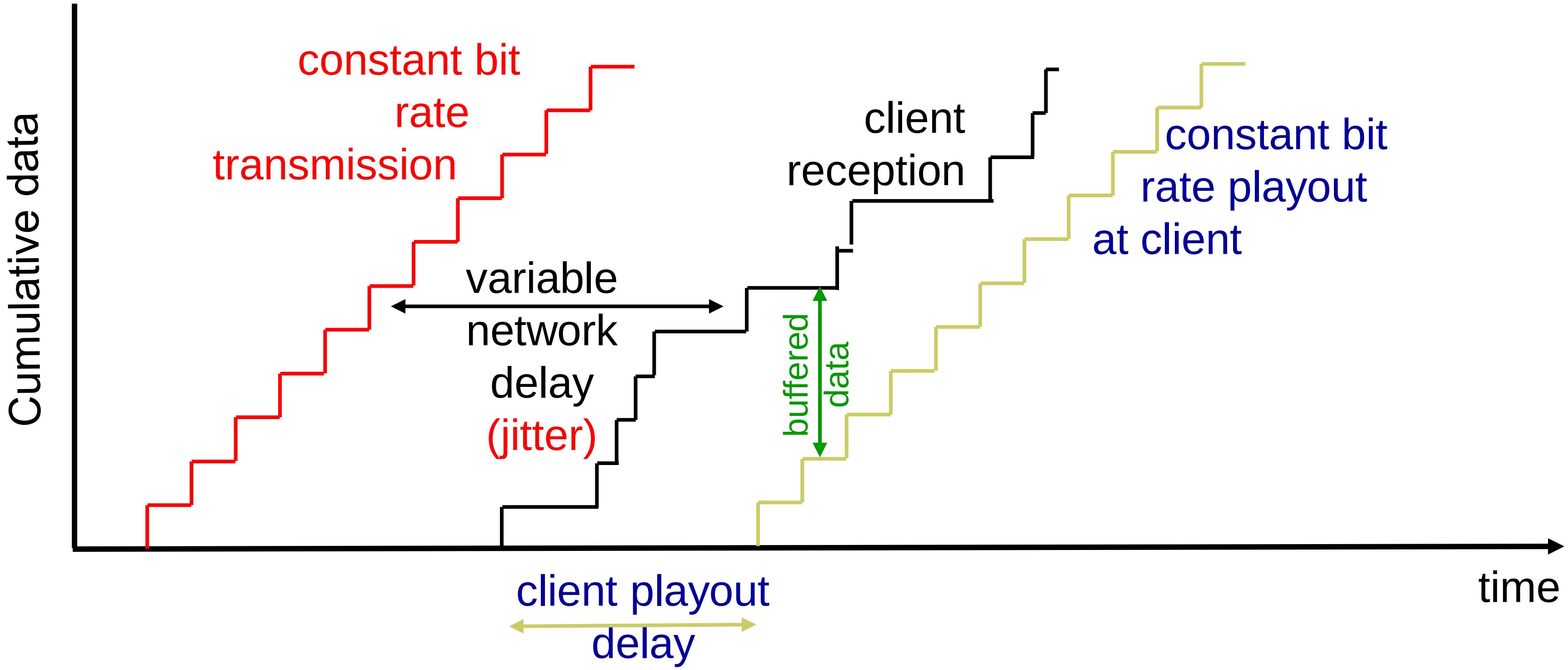
VoIP: packet loss, delay

- *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, loss concealment, packet loss rates between 1% and 10% can be tolerated

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Delay jitter

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN



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

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

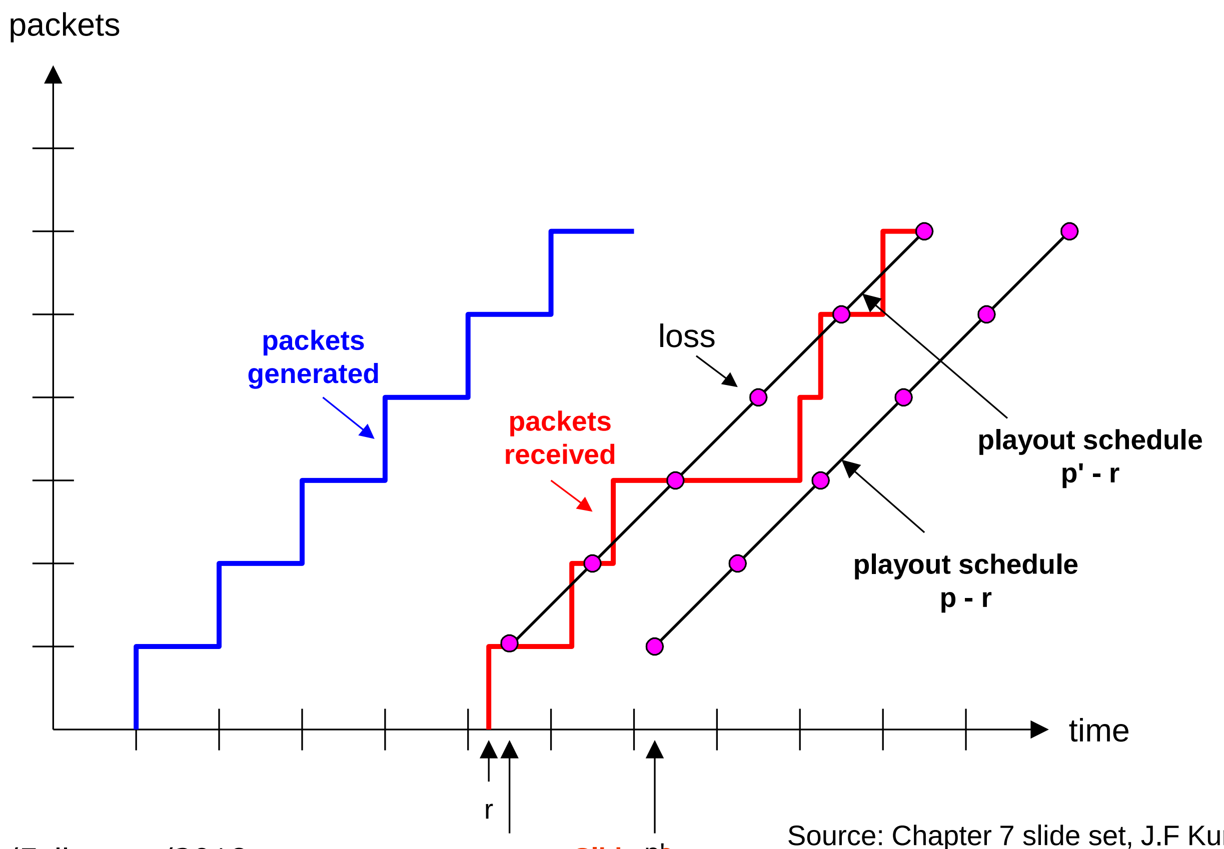
VoIP: fixed playout delay

- 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 in choosing *q*:
  - *large q:* less packet loss
  - *small q:* better interactive experience

VoIP: fixed playout delay

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

- 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'*



Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Adaptive playout delay (1)

- *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):

$$d_i = (1-\alpha)d_{i-1} + \alpha\ \underbrace{(r_i - t_i)}_{\text{measured delay of ith packet}}$$

delay estimate after ith packet

small constant, e.g. 0.1

time received - time sent (timestamp)

measured delay of ith packet

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Adaptive playout delay (2)

- also useful to estimate average deviation of delay, v_i

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

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

  - for first packet in talk spurt, play-out time is:

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

- remaining packets in talk spurt are played out periodically

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Adaptive playout delay (3)

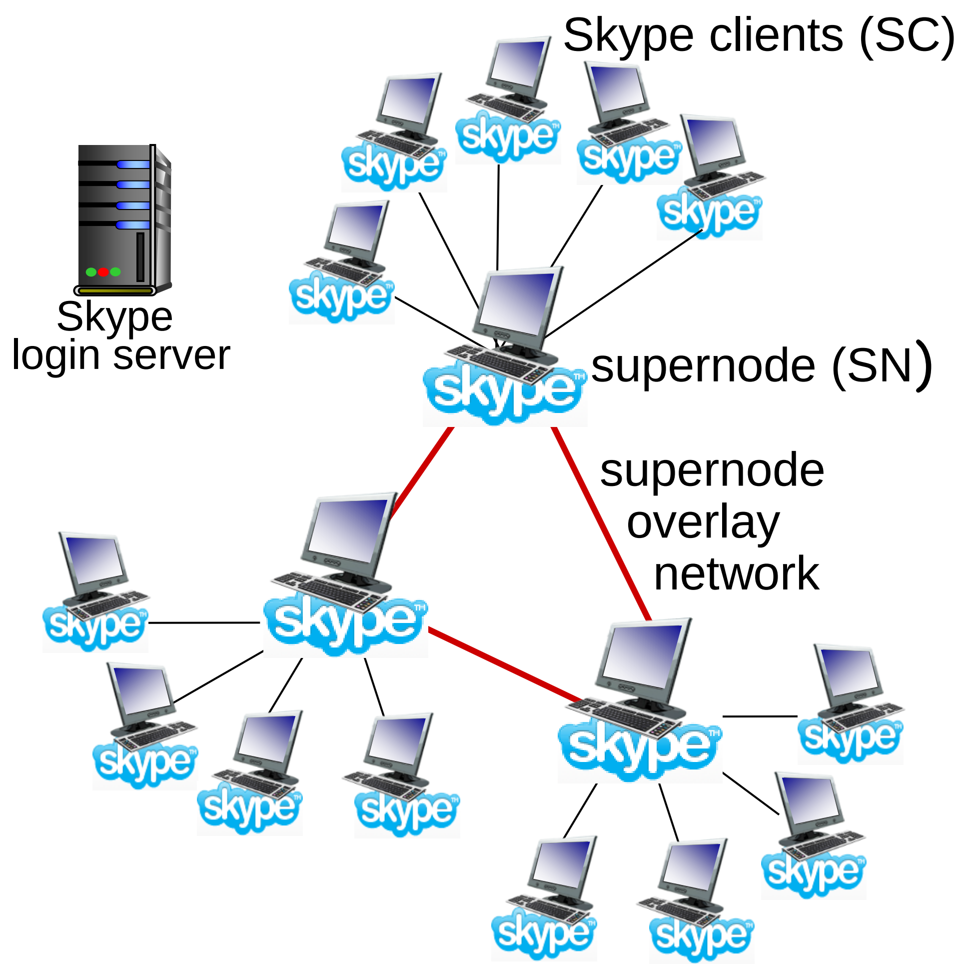
Q: How does receiver determine whether packet is first in a talkspurt?

- 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.

Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Voice-over-IP: Skype

- proprietary application-layer protocol (inferred via reverse engineering)
  - encrypted msgs
- 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

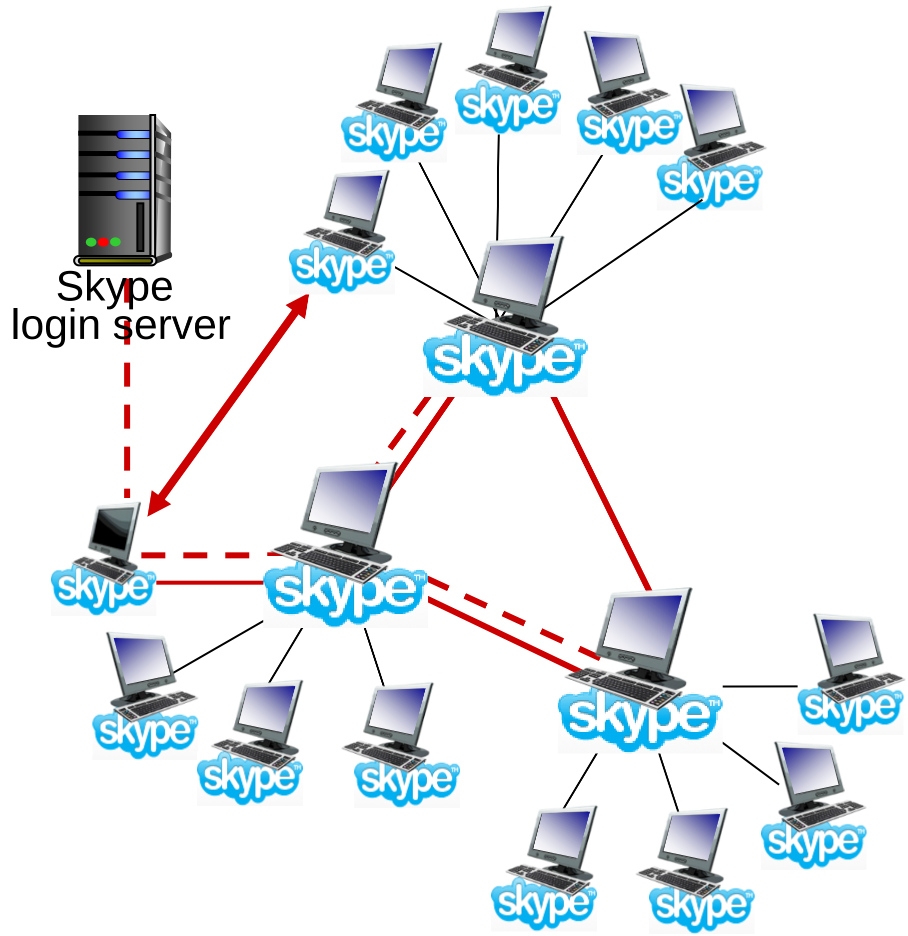


Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

P2P voice-over-IP: skype

skype client operation:

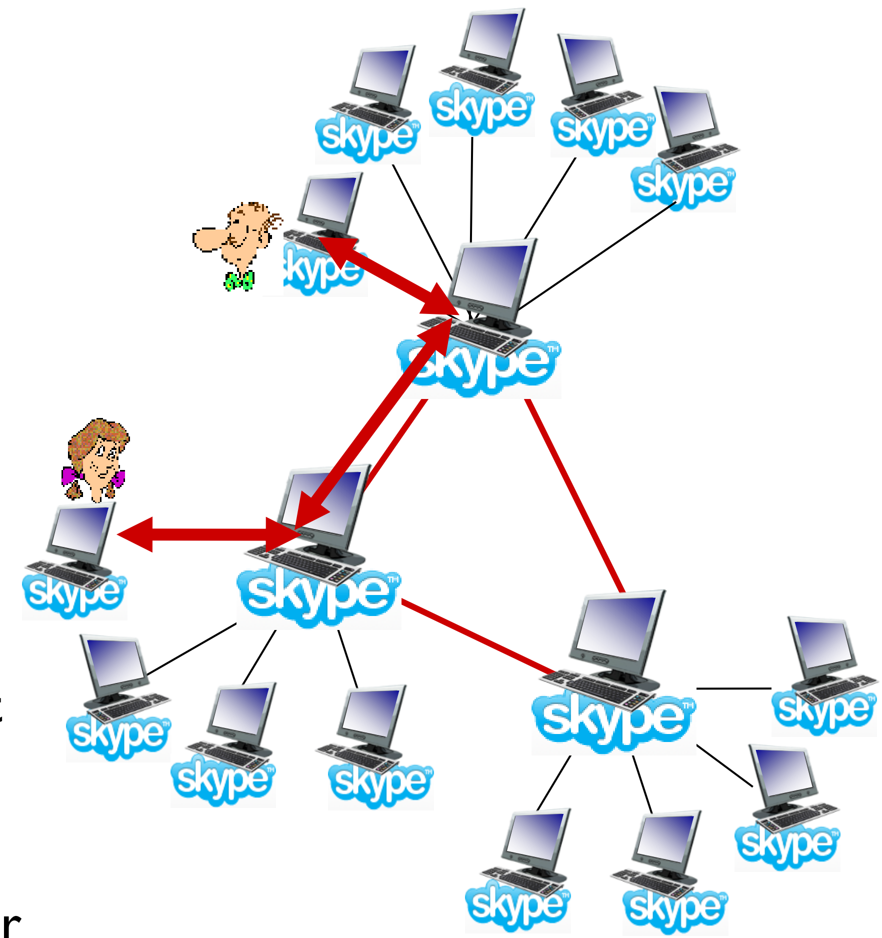
1. joins skype network by contacting SN (IP address cached) using TCP
2. logs-in (usename, password) to centralized skype login server
3. obtains IP address for callee from SN, SN overlay
   - or client buddy list
4. initiate call directly to callee



Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

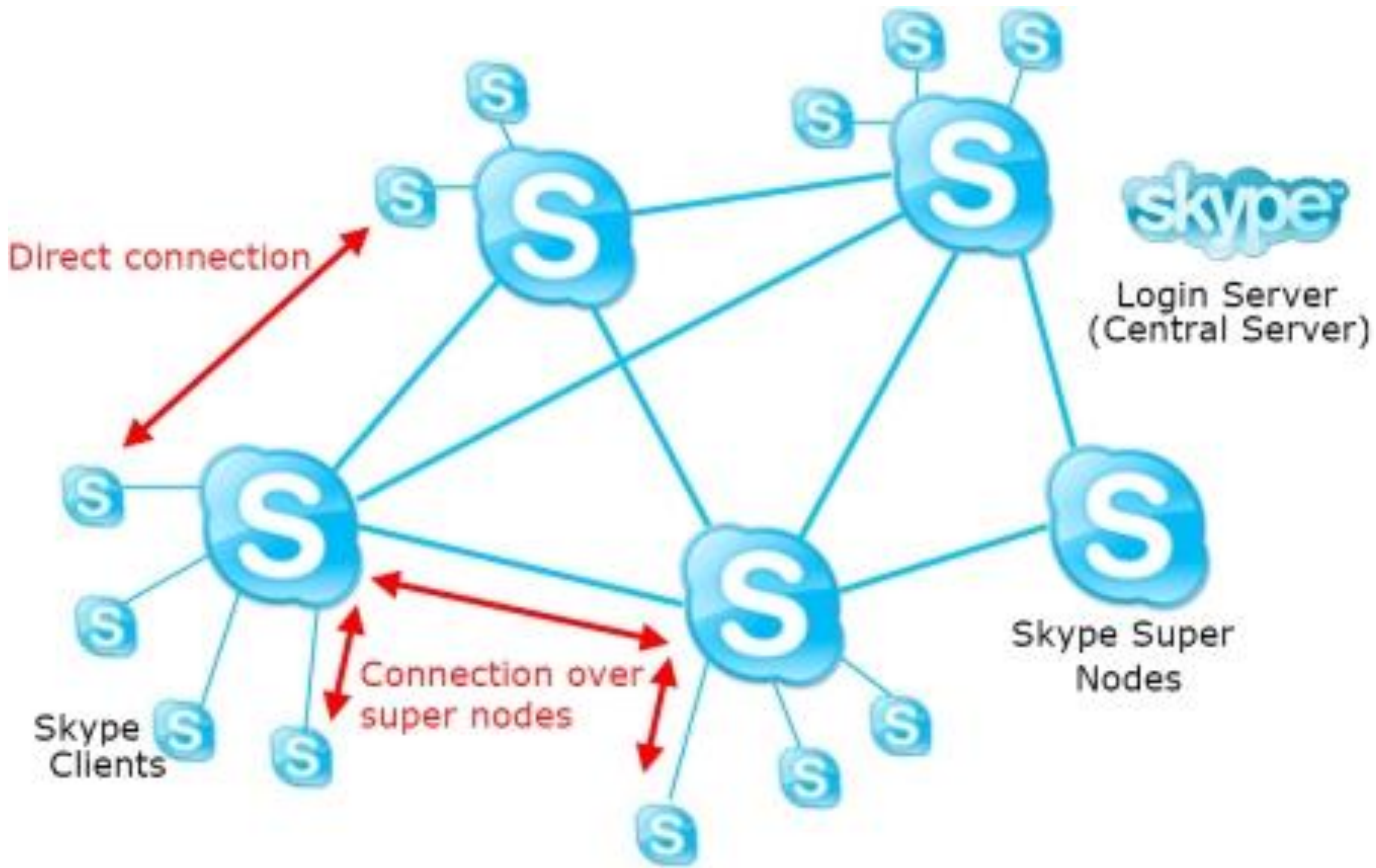
Skype: peers as relays

- *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



Source: Chapter 7 slide set, J.F Kurose and K.W. Ross, Ed 6

Skype Architecture



Source: http://letsbytecode.com

Conclusion

- We discussed network performance in detail
  - Why load is dynamic and unpredictable
  - Outsourcing the problem to CDNs
  - CDN architecture
  - Other architectures: Skype
- Next Lecture:
  - Quality-of-Service
  - Net Neutrality