Logistics

- HW2 is posted and due in 2 weeks
- HW1sol will be posted tonight
- I-clicker starts being recorded now

I-clicker Question

Q1: Consider the following DNS RR :
(networkutopia.com, dns1.networkutopia.com)
Which of the following statements is true:

☐ A: This is a type A record

✓ B: This record is necessary for the networkutopia.com domain to be reached from the outside world

☐ C: This record is sufficient for networkutopia.com to be reached from the outside world

☐ D: This record is stored at the authoritative DNs server of networkutopia.com

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks

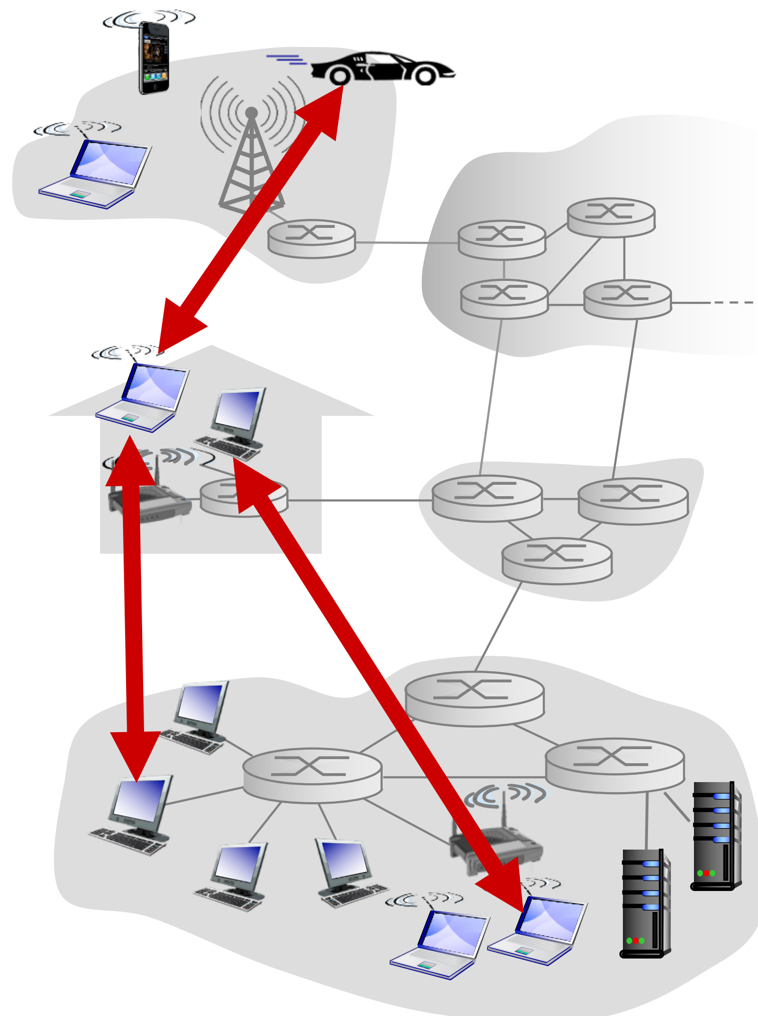
2.7 socket programming with UDP and TCP

Pure P2P architecture

- *no* always-on server
- arbitrary end systems directly communicate
- peers are intermittently connected and change IP addresses

examples:

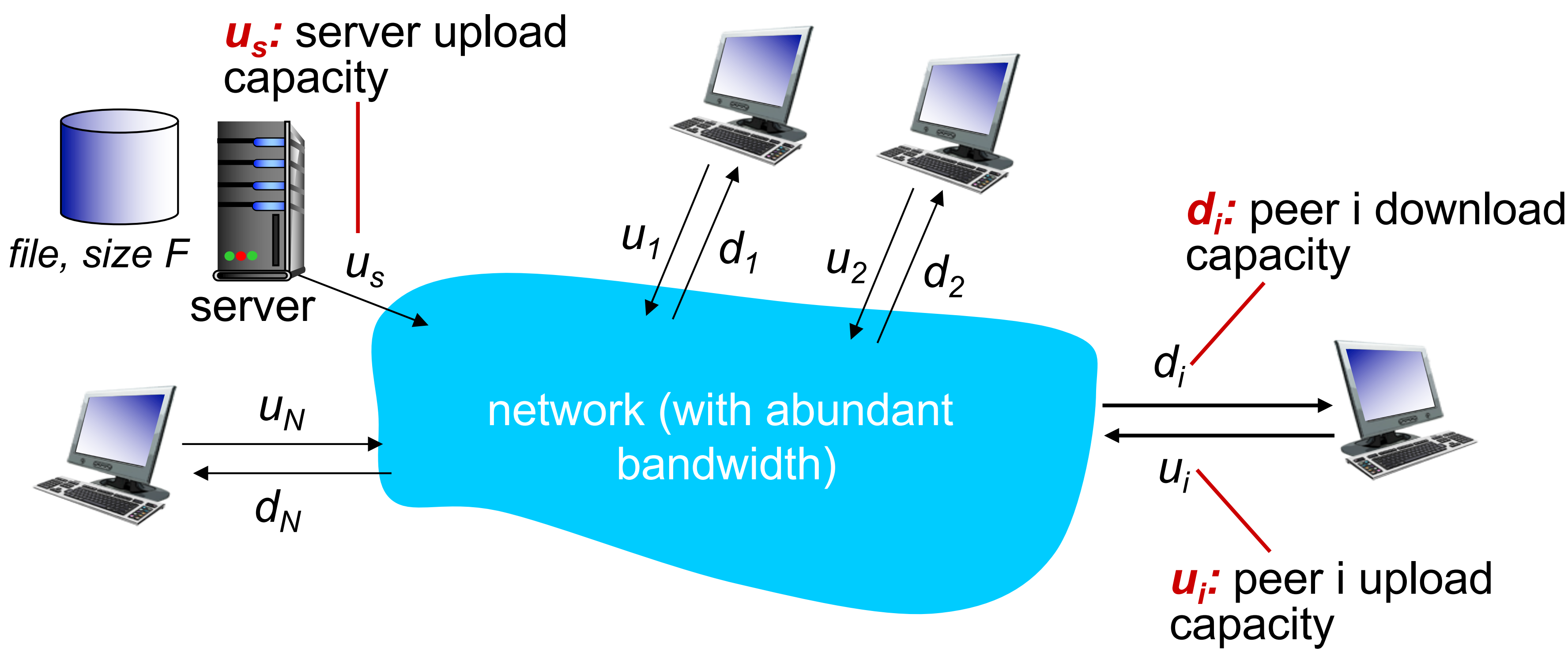
- file distribution (BitTorrent)
- Streaming (KanKan)
- VoIP (Skype)



File distribution: client-server vs P2P

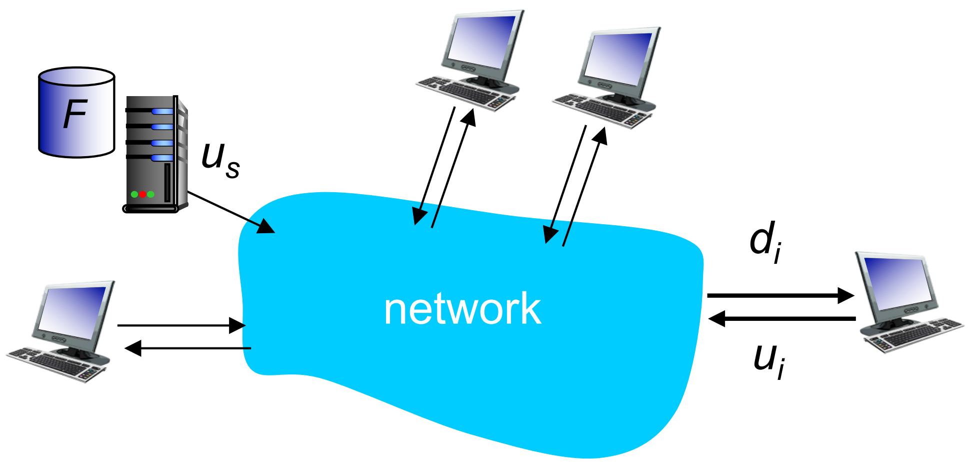
Question: how much time to distribute file (size F) from one server to N peers?

- peer upload/download capacity is limited resource



File distribution time: client-server

- *server transmission:* must sequentially send (upload) N file copies:
  - time to send one copy: F/u_s
  - time to send N copies: NF/u_s
- *client:* each client must download file copy
  - d_{min} = min client download rate
  - min client download time: F/d_{min}



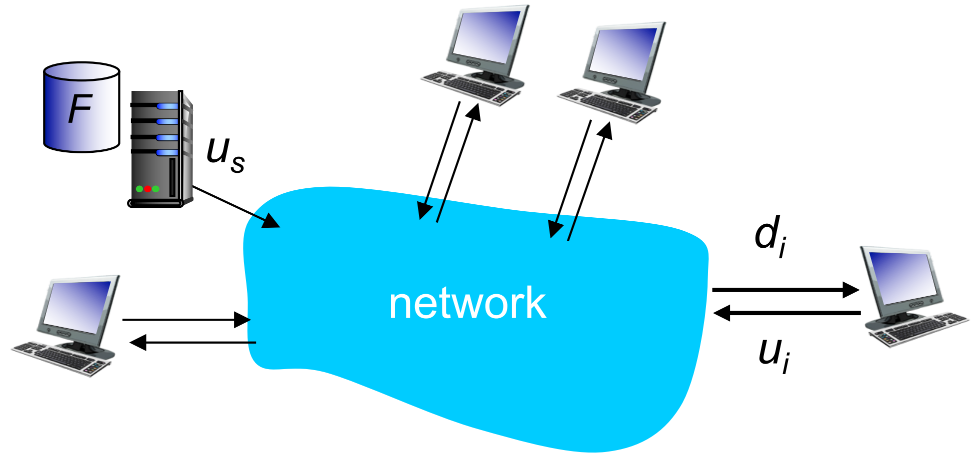
time to distribute F to N clients using client-server approach

$$D_{c\text{-}s} \geq \max\{NF/u_s, F/d_{min}\}$$

increases linearly in N

File distribution time: P2P

- *server transmission:* must upload at least one copy
  - time to send one copy: F/u_s
- *client:* each client must download file copy
  - min client download time: F/d_{min}
- *clients:* as aggregate must download *NF* bits
  - max upload rate (limiting max download rate) is $u_s + \Sigma u_i$



time to distribute F to N clients using P2P approach

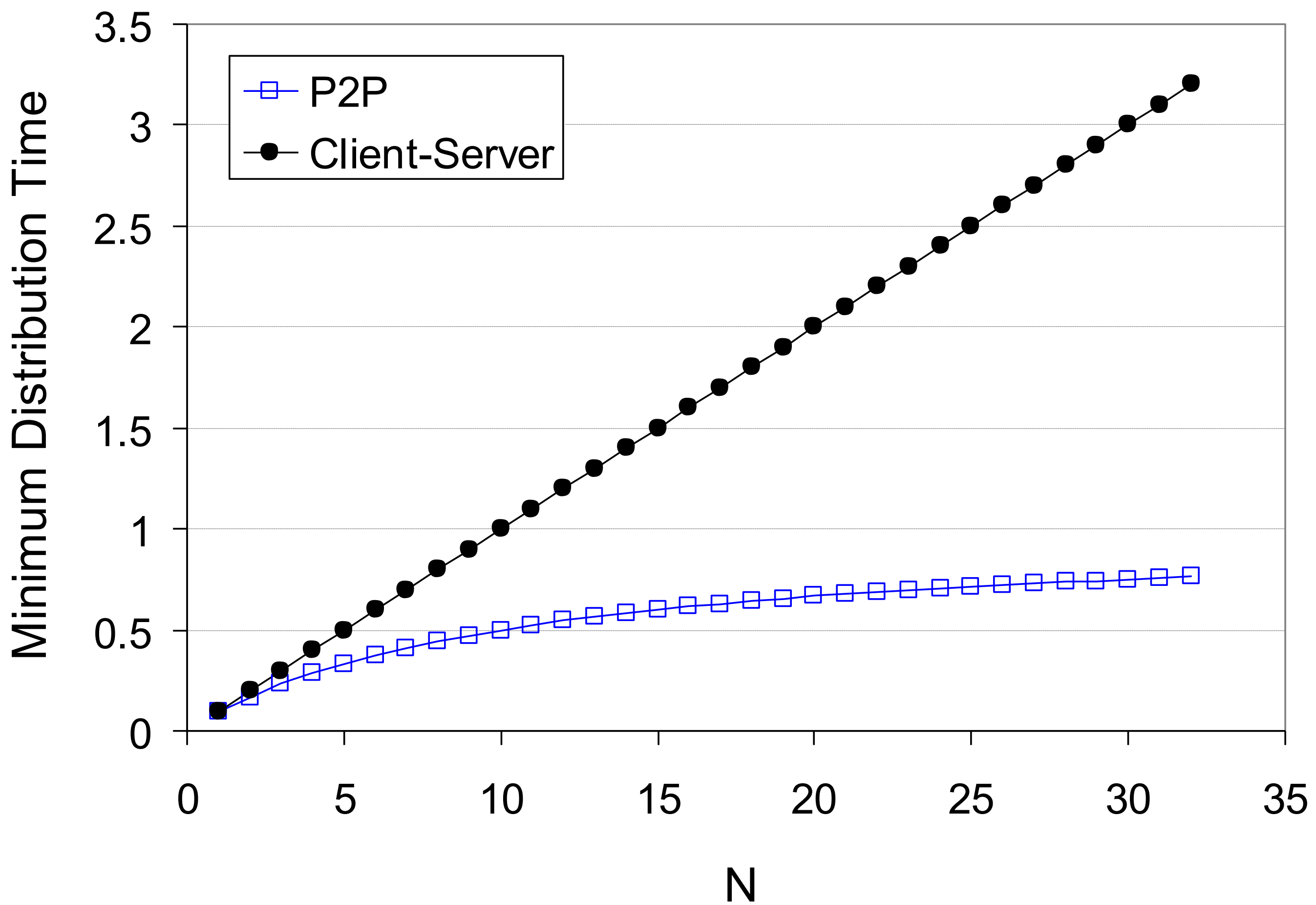
$$D_{P2P} \geq max\{F/u_s,F/d_{min},NF/(u_s + \Sigma u_i)\}$$

increases linearly in *N* …

… but so does this, as each peer brings service capacity

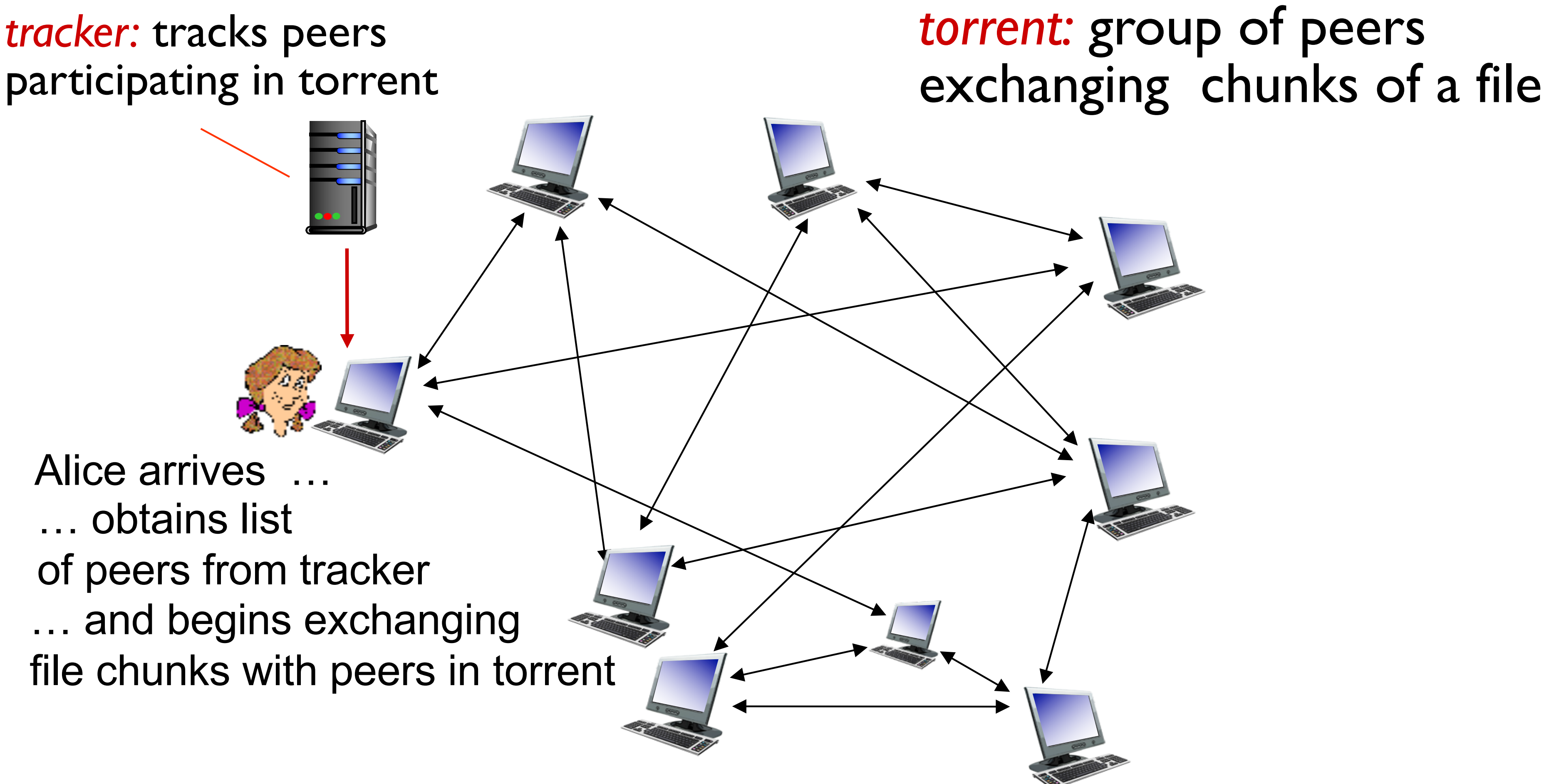
Client-server vs. P2P: example

client upload rate = u, F/u = 1 hour, $u_s = 10u$, $d_{min} \geq u_s$



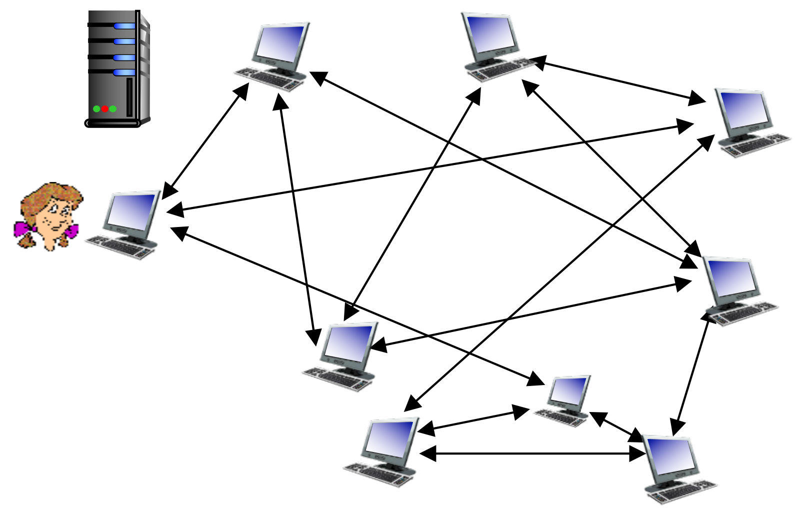
P2P file distribution: BitTorrent

- file divided into 256Kb chunks
- peers in torrent send/receive file chunks



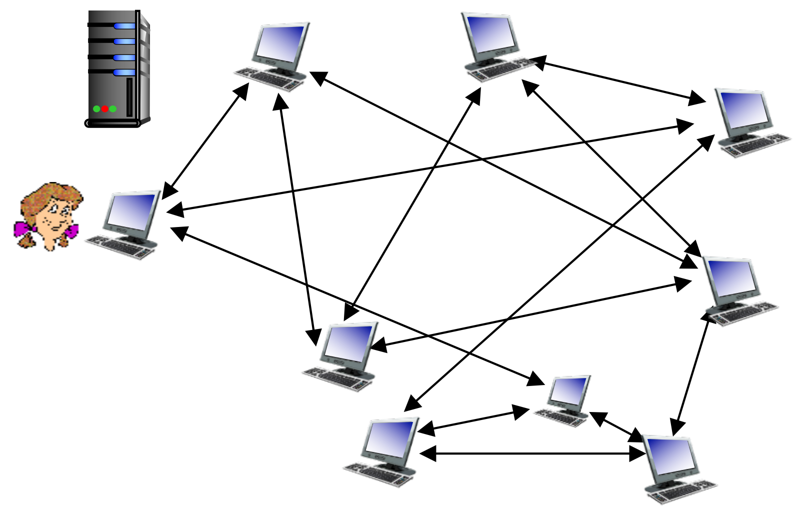
P2P file distribution: BitTorrent

- peer joining torrent:
  - has no chunks, but will accumulate them over time from other peers
  - registers with tracker to get list of peers, connects to subset of peers ("neighbors")



- while downloading, peer uploads chunks to other peers
- peer may change peers with whom it exchanges chunks
- *churn:* peers may come and go
- once peer has entire file, it may (selfishly) leave or (altruistically) remain in torrent

P2P file distribution: BitTorrent



Two design decisions for each peer:

1. Topology management: To which subset of neighbors to connect
   - Good peers (good for me – selfish) vs. not so good peers (good for them – altruistic) vs discover new peers
2. Scheduling: Which chunk to download from those neighbors
   - Bitorrent: Rarest block first – see Coupon Collector Problem: https://en.wikipedia.org/wiki/Coupon_collector%27s_problem
   - Streaming: also take into account order

BitTorrent: requesting, sending file chunks

requesting chunks:

- at any given time, different peers have different subsets of file chunks
- periodically, Alice asks each peer for list of chunks that they have
- Alice requests missing chunks from peers, rarest first

sending chunks: tit-for-tat

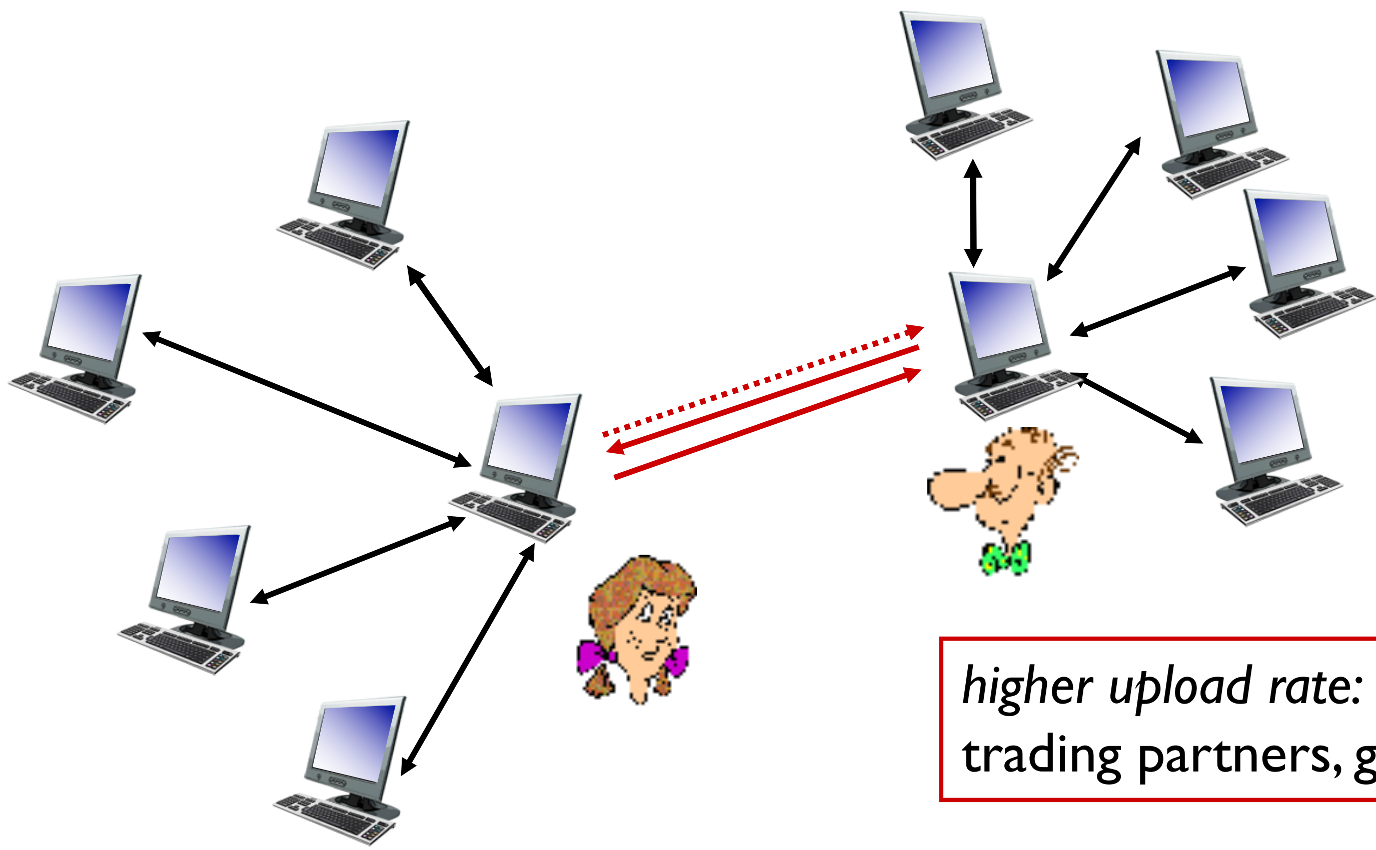
- Alice sends chunks to those four peers currently sending her chunks *at highest rate*
  - other peers are choked by Alice (do not receive chunks from her)
  - re-evaluate top 4 every10 secs
- every 30 secs: randomly select another peer, starts sending chunks
  - "optimistically unchoke" this peer
  - newly chosen peer may join top 4

BitTorrent: tit-for-tat

(1) Alice "optimistically unchokes" Bob

(2) Alice becomes one of Bob's top-four providers; Bob reciprocates

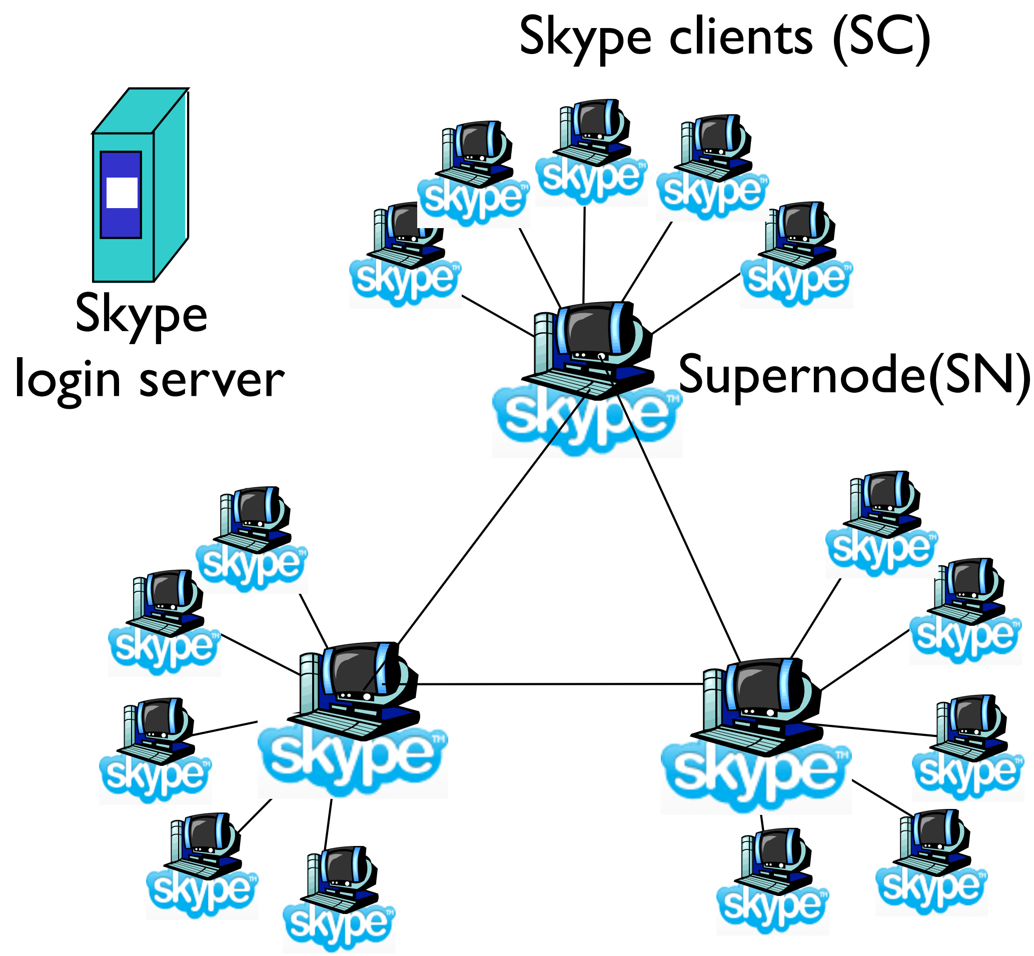
(3) Bob becomes one of Alice's top-four providers



higher upload rate: find better trading partners, get file faster !

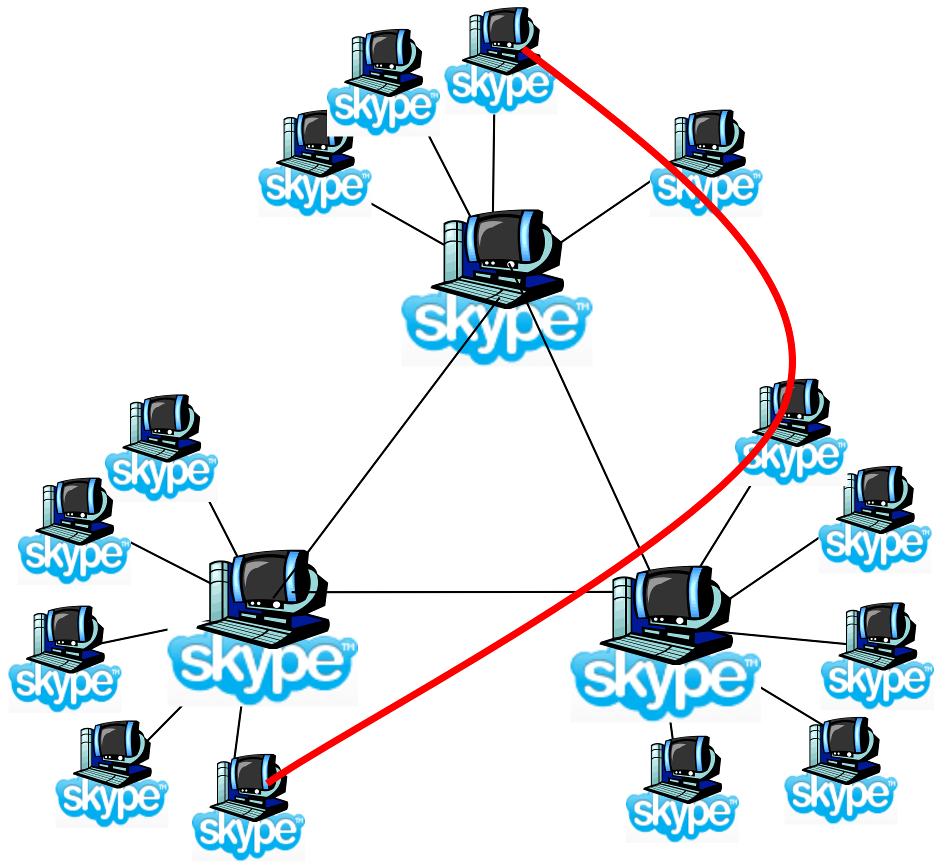
P2P Case study: Skype

- inherently P2P: pairs of users communicate.
- proprietary application-layer protocol (inferred via reverse engineering)
- hierarchical overlay with Supernode (SNs)
- User location
  - Index maps usernames to IP addresses; distributed over SNs



Peers as relays

- problem when both Alice and Bob are behind "NATs".
  - NAT prevents an outside peer from initiating a call to insider peer
- solution:
  - using Alice's and Bob's SNs, *relay* is chosen
  - each peer initiates session with relay.
  - peers can now communicate through NATs via relay



I-clicker Question

Q2: Consider a P2P application, such as BitTorrent: Which of the following statements is false:

✓A: P2P need a different kind of socket than typical client-server applications

☐ B: P2P differ from typical client-server applications in that the service is provided jointly among the peers rather than provided by the server(s)

☐ C: Not all P2P technologies are illegal

☐ D: P2P technologies scale better than client-server

Chapter 2: outline

2.1 principles of network applications

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

2.5 P2P applications

2.6 video streaming and content distribution networks (CDNs)

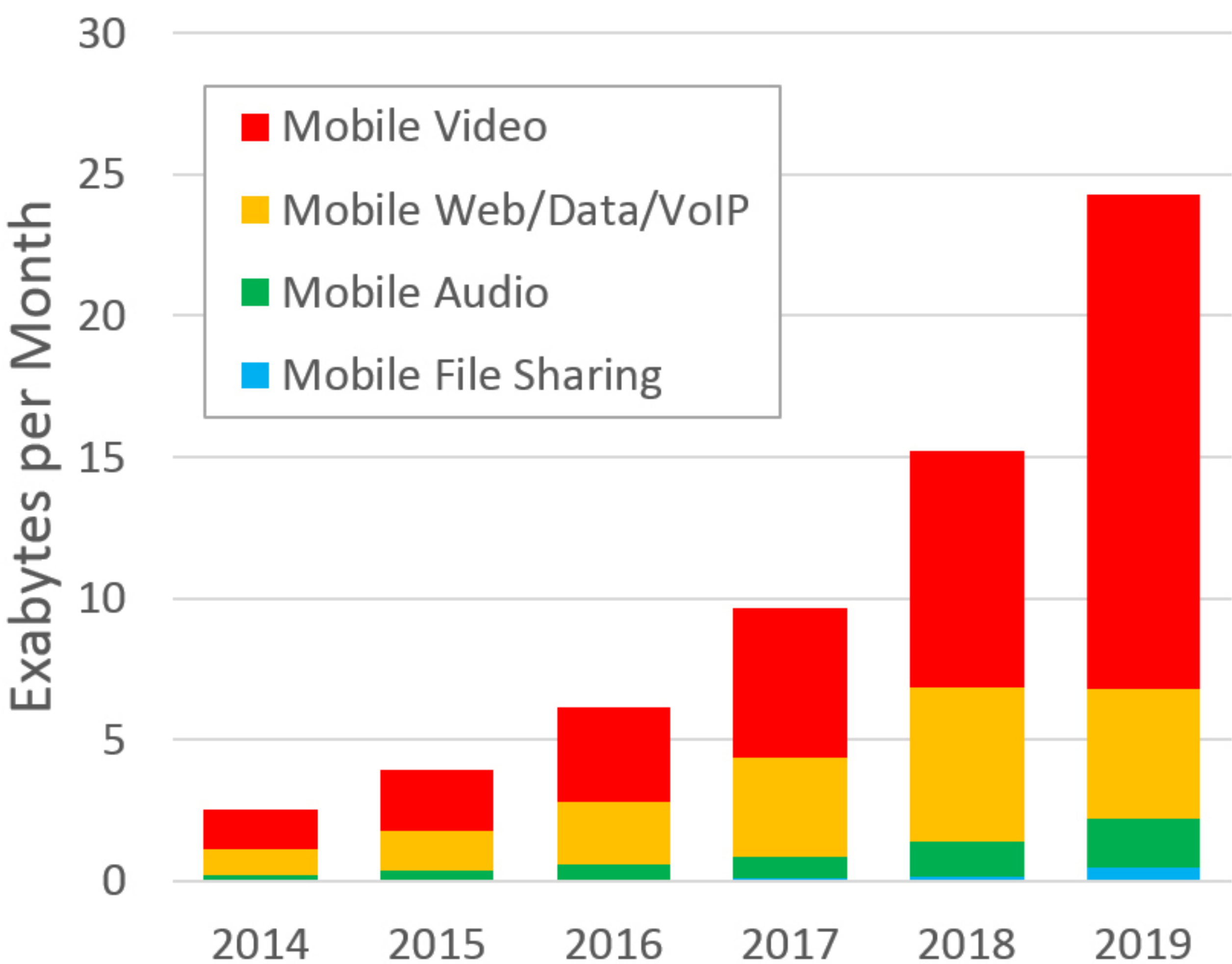
more video: Ch.9

2.7 socket programming with UDP and TCP

Video Traffic Dominates

http://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html

"82% of all IP traffic will be video by 2021"



Cisco Mobile VNI 2014-2019

Video Streaming and CDNs: context

- Video Traffic: major consumer of Internet bandwidth
  - Netflix, YouTube: 37%, 16% of downstream residential ISP traffic
  - ~1B YouTube users, ~75M Netflix users
- Challenges
  - scale - how to reach ~1B users?
  - heterogeneity
    - different users have different capabilities
      - (wired vs. mobile; bandwidth rich vs. poor)
  - dynamic rate and bandwidth
  - video is a special traffic type.
- *Video Delivery Today:* distributed, application-lev infrastructure:

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*

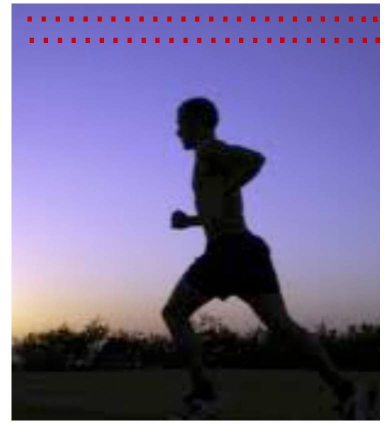
temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame *i+1*

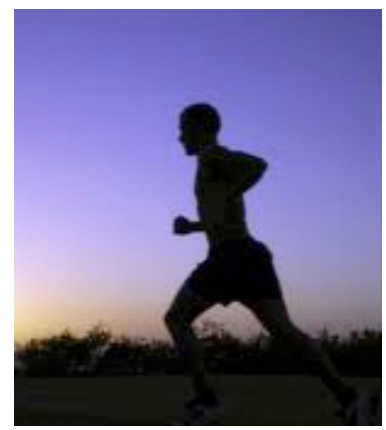
Multimedia: video

- 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*

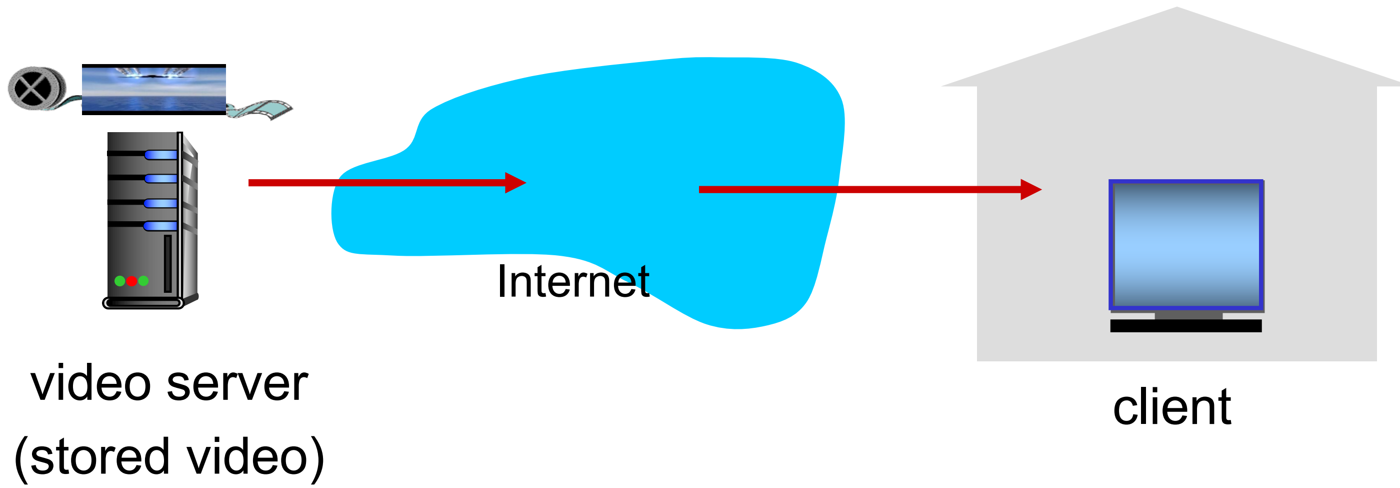


frame *i+1*

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

Streaming stored video:

simple scenario:



Adaptive Video Streaming Standards: [Ch.9]

MPEG-DASH, Apple's HLS, Microsoft's Smooth Streaming

Streaming multimedia: MPEG-DASH

- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
  - https://en.wikipedia.org/wiki/Dynamic_Adaptive_Streaming_over_HTTP
- *server:*
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file:* provides URLs for different chunks
- *client:*
  - monitors buffer occupancy
  - 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)

Streaming multimedia: DASH

- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
- *"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)

Content distribution networks

- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of *simultaneous* users?

- *option 1:* single, large "mega-server"
  - single point of failure
  - point of network congestion
  - long path to distant clients
  - multiple copies of video sent over outgoing link

….quite simply: this solution *doesn't scale*

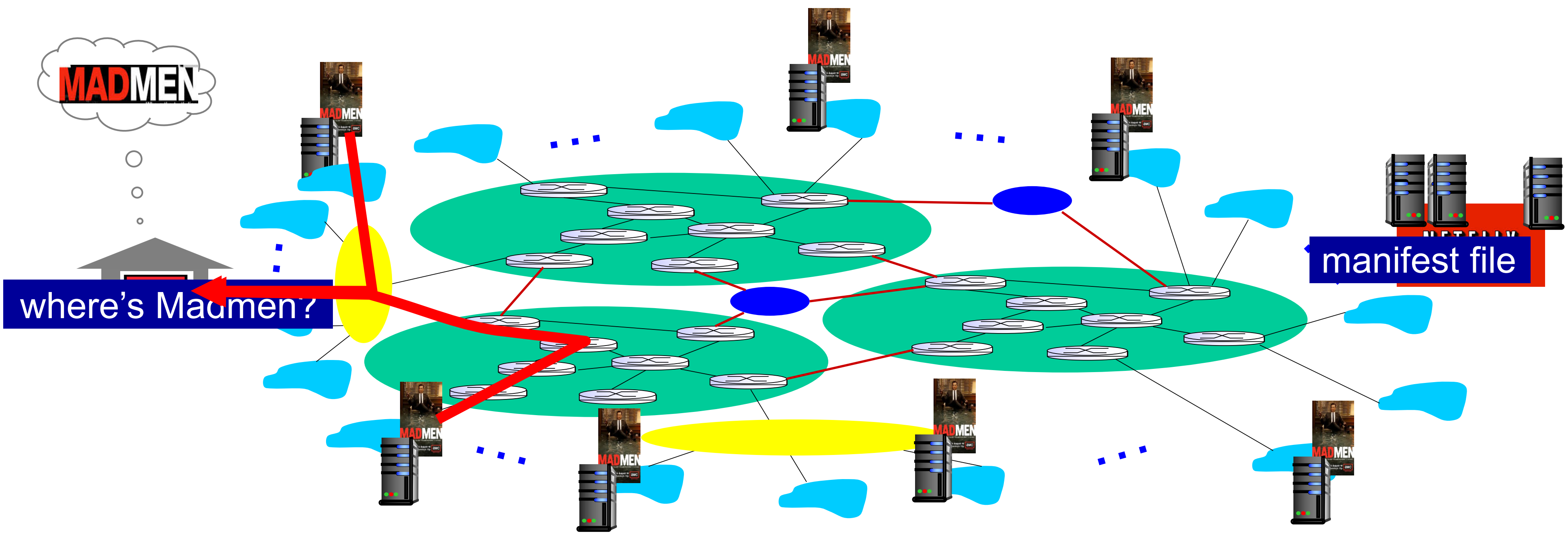
Content distribution networks

- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

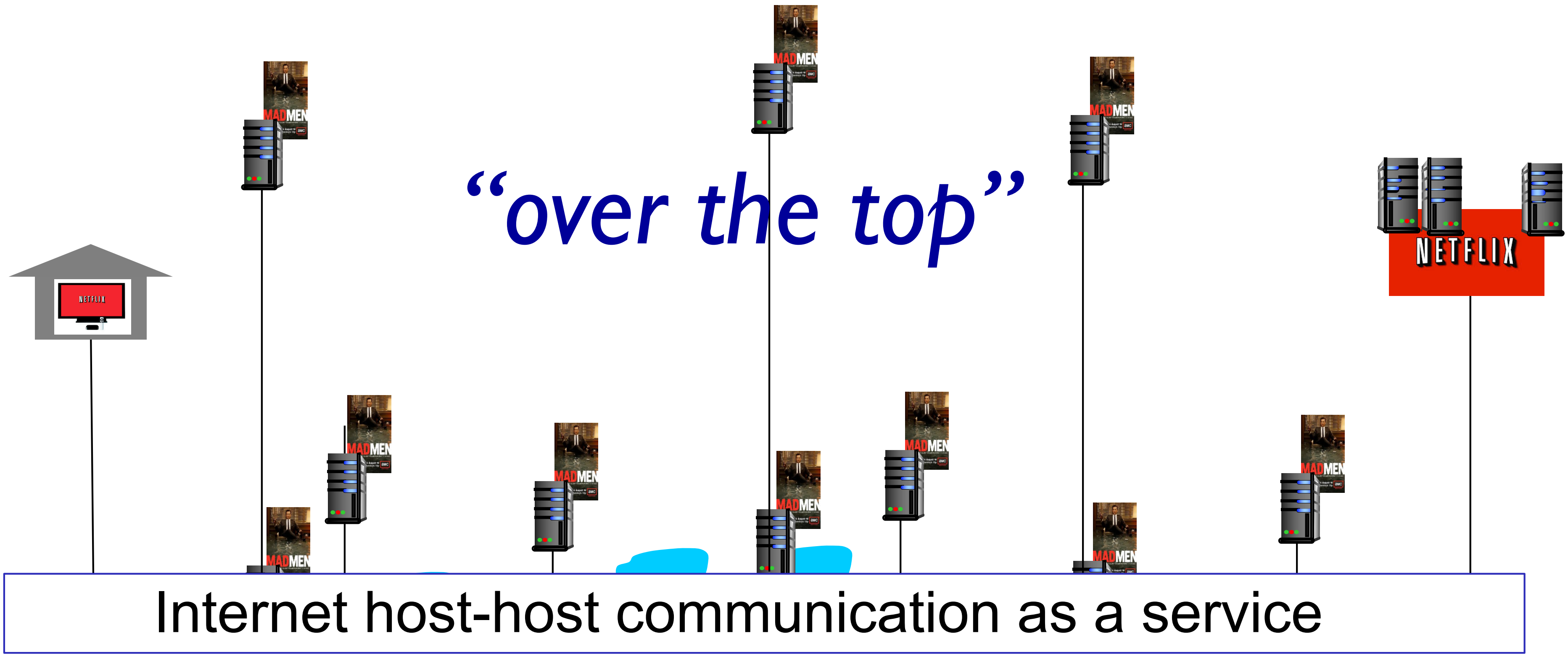
- *option 2:* store/serve multiple copies of videos at multiple geographically distributed sites *(CDN)*
  - *enter deep:* push CDN servers deep into many access networks
    - close to users
    - used by Akamai, 1700 locations
  - *bring home:* smaller number (10's) of larger clusters in POPs near (but not within) access networks
    - used by Limelight

Content Distribution Networks (CDNs)

- Third party CDNs, e.g. Akamai: 100,000+ servers in 1000+ clusters in 1000+ networks in 70+ countries serving trillions of requests a day.
- Private CDNs: Google's CDN, etc…
- CDN: stores copies of content at CDN nodes
  - e.g. Netflix stores copies of MadMen
- subscriber requests content from CDN
  - directed to nearby copy, retrieves content
  - may choose different copy if network path congested



Content Distribution Networks (CDNs)



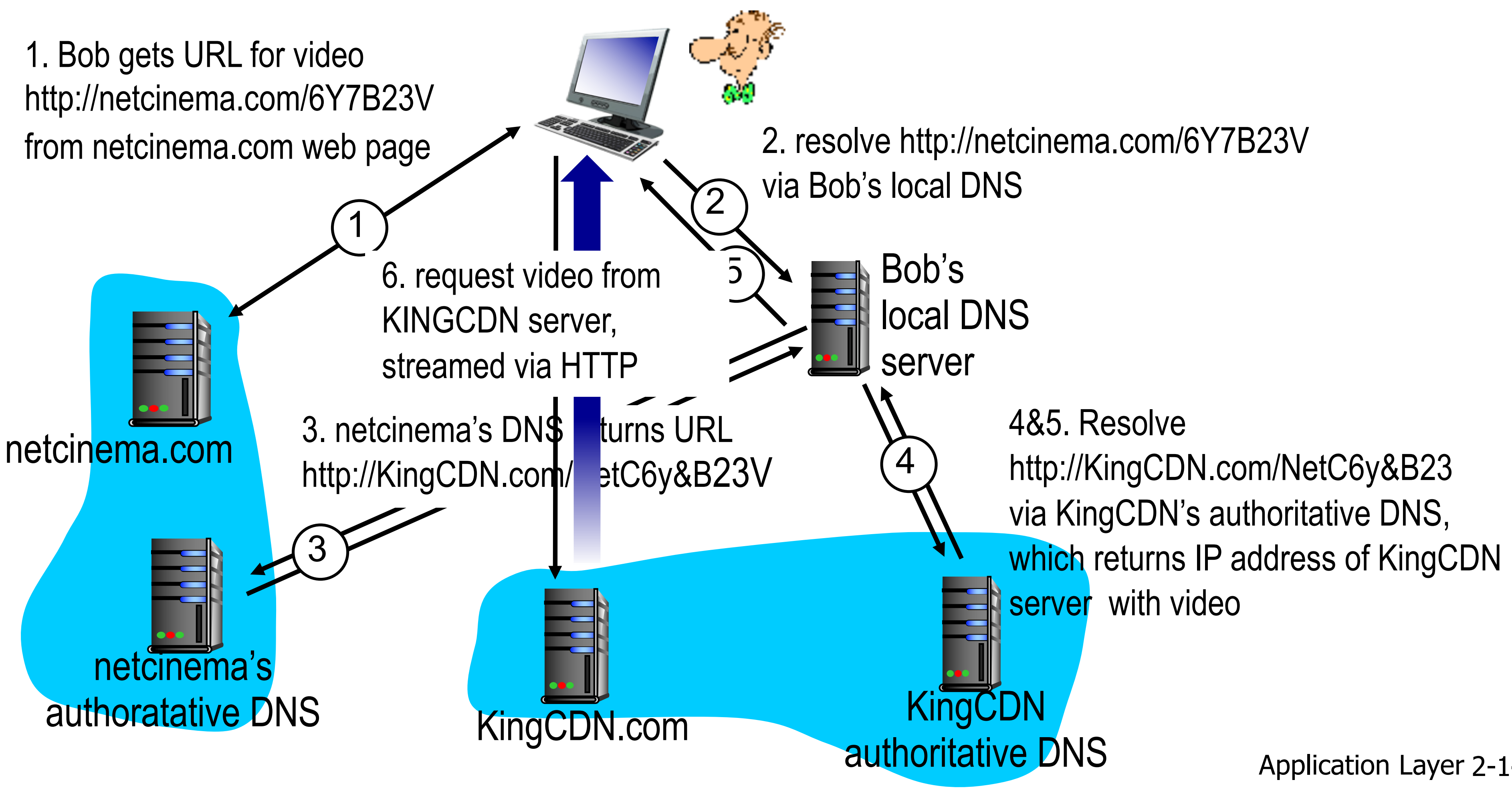
OTT challenges: coping with a congested Internet

- from which CDN node to retrieve content?
- viewer behavior in presence of congestion?
- what content to place in which CDN node?

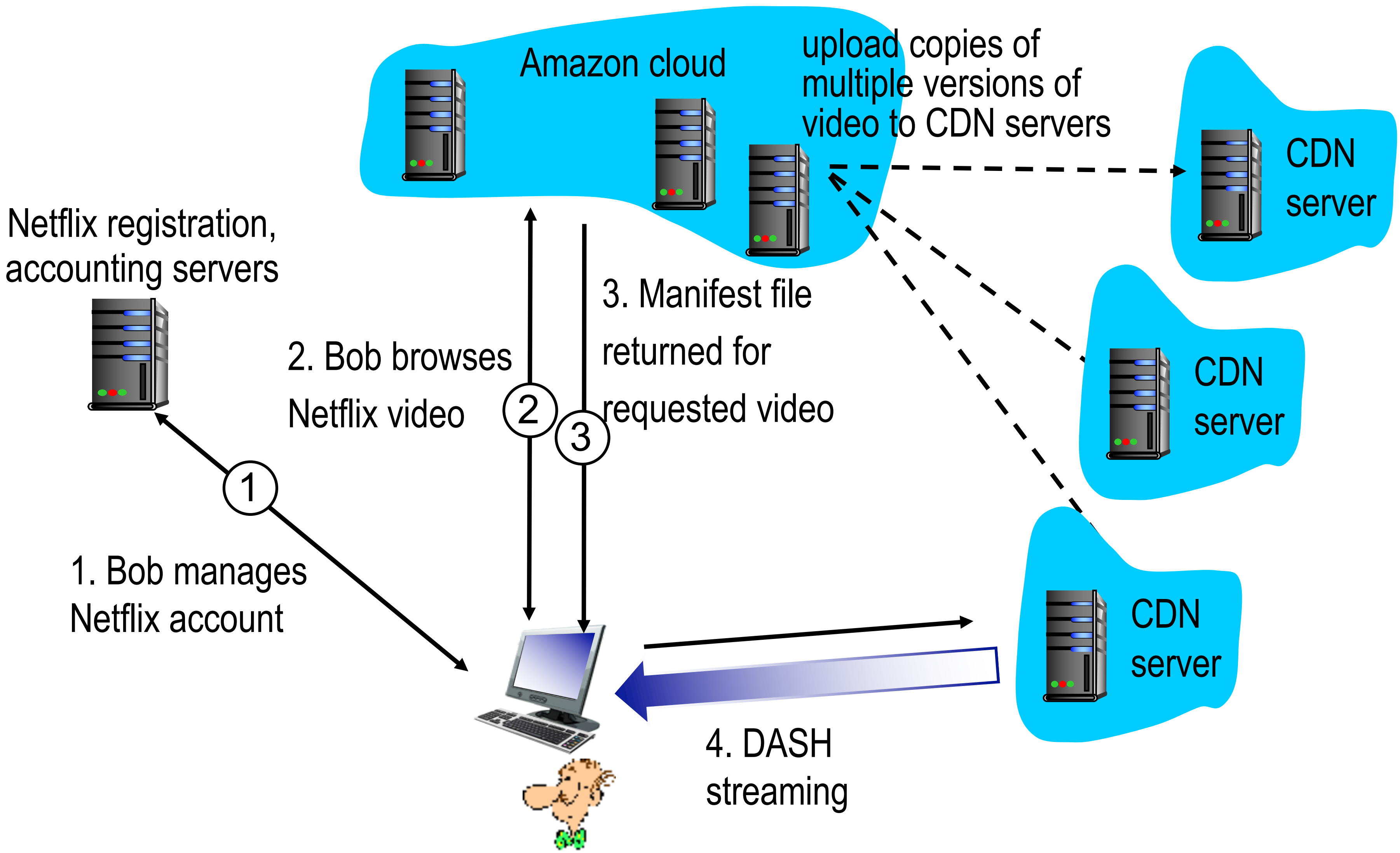
CDN content access: a closer look

Bob (client) requests video http://netcinema.com/6Y7B23V

- video stored in CDN at http://KingCDN.com/NetC6y&B23V



Case study: Netflix



I-clicker Question

Q3: "Video Streaming is different than Data (HTTP, FTP) applications because….."

- ☐ A: Video packets have dependencies due to encoding
- ☐ B: Video streaming has some delay requirements
- ☐ C: Video does not need 100% reliability
- ✓ D: All of the above