Chapter 2
Application Layer

Acknowledgement: Based on the textbook's website:
  https://gaia.cs.umass.edu/kurose_ross/index.php
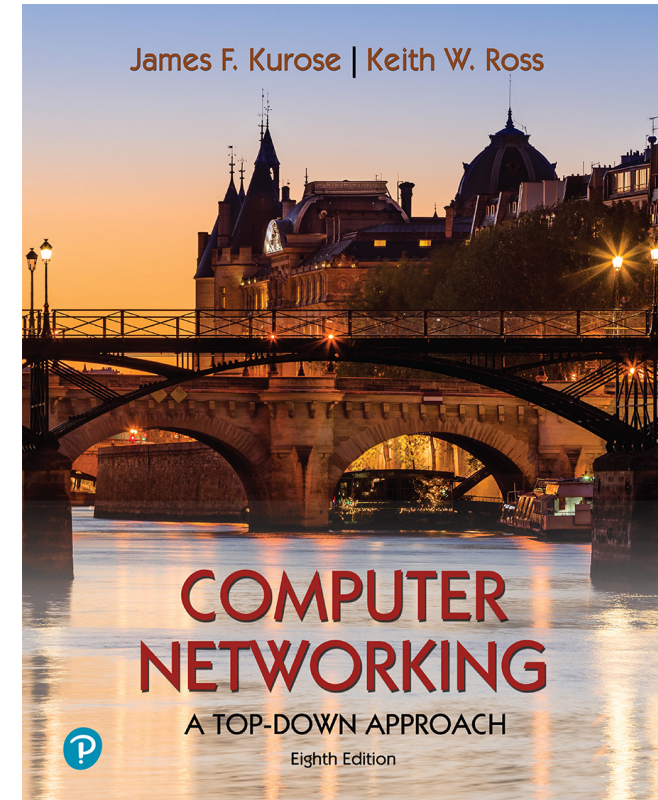
Computer Networking: A Top-Down Approach

8th edition   n
Jim Kurose, Keith Ross
Pearson, 2020

Application Layer: Overview

- Principles of network applications

- Web and HTTP

- E-mail, SMTP, IMAP

- The Domain Name System DNS

- **P2P applications**

- video streaming and content distribution networks

- socket programming with UDP and TCP
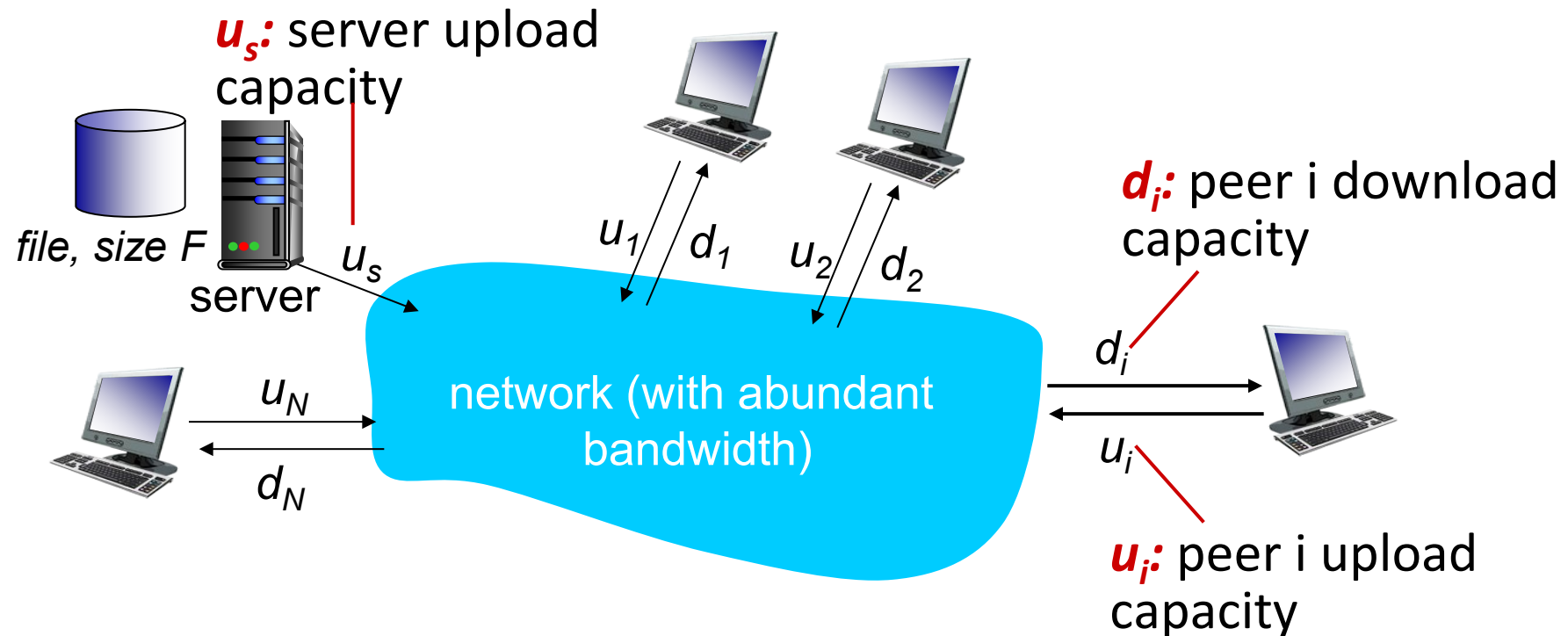
Peer-to-peer (P2P) architecture

- *no* always-on server
- arbitrary end systems directly communicate
- peers request service from other peers, provide service in return to other peers
  - *self scalability* – new peers bring new service capacity, and new service demands
- peers are intermittently connected and change IP addresses
  - complex management
- examples: P2P file sharing (BitTorrent), streaming (KanKan), VoIP (Skype)
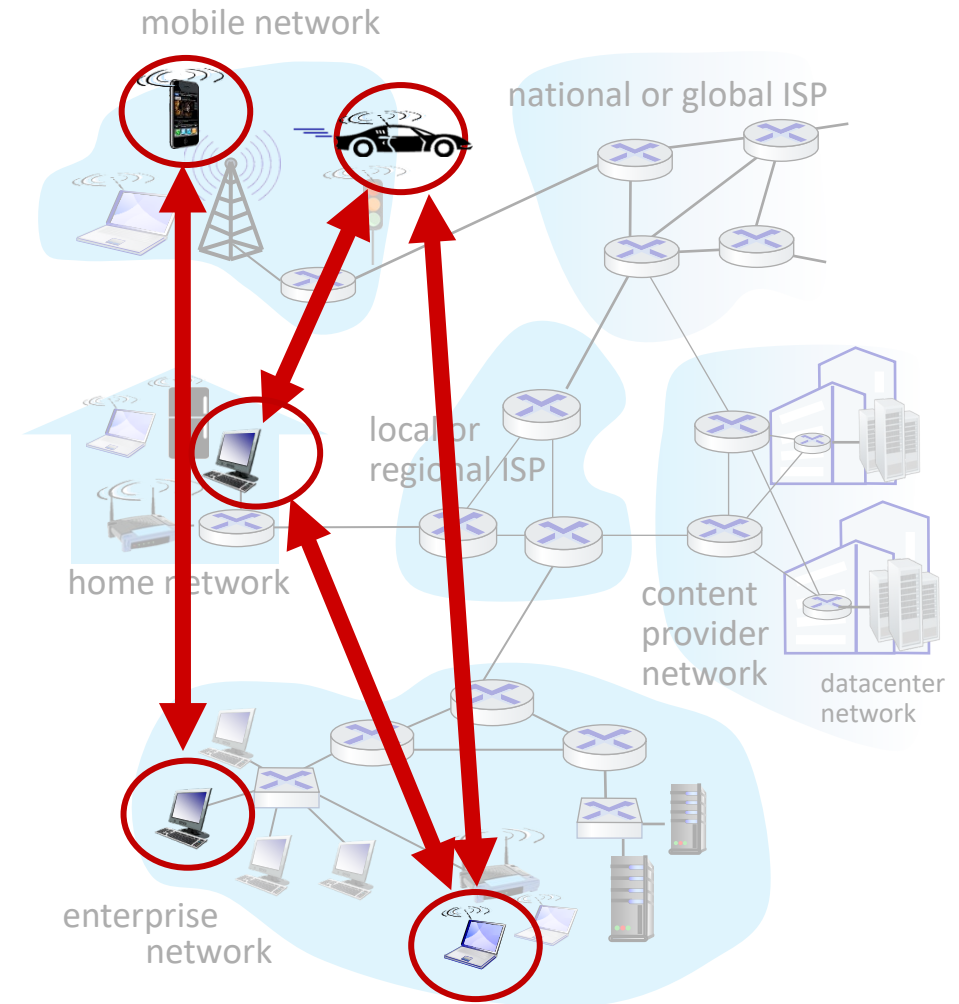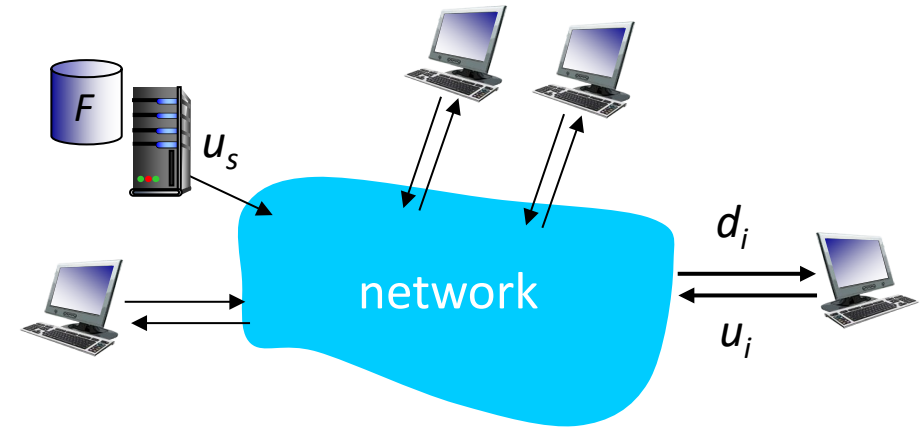
File distribution: client-server vs P2P

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

- peer upload/download capacity is limited resource

u_s: server upload capacity

d_i: peer i download capacity

u_i: peer i upload capacity

file, size F

server

u_s

u_1 d_1 u_2 d_2

u_N

d_N

network (with abundant bandwidth)

d_i

u_i

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
 - max client download time: F/d_{min}

$$\text{time to distribute } F \text{ to } N \text{ clients using client-server approach} \quad D_{c\text{-}s} \geq max\{NF/u_s, F/d_{min}\}$$

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
  - max client download time: F/d_{min}

- *clients:* as aggregate must download NF bits
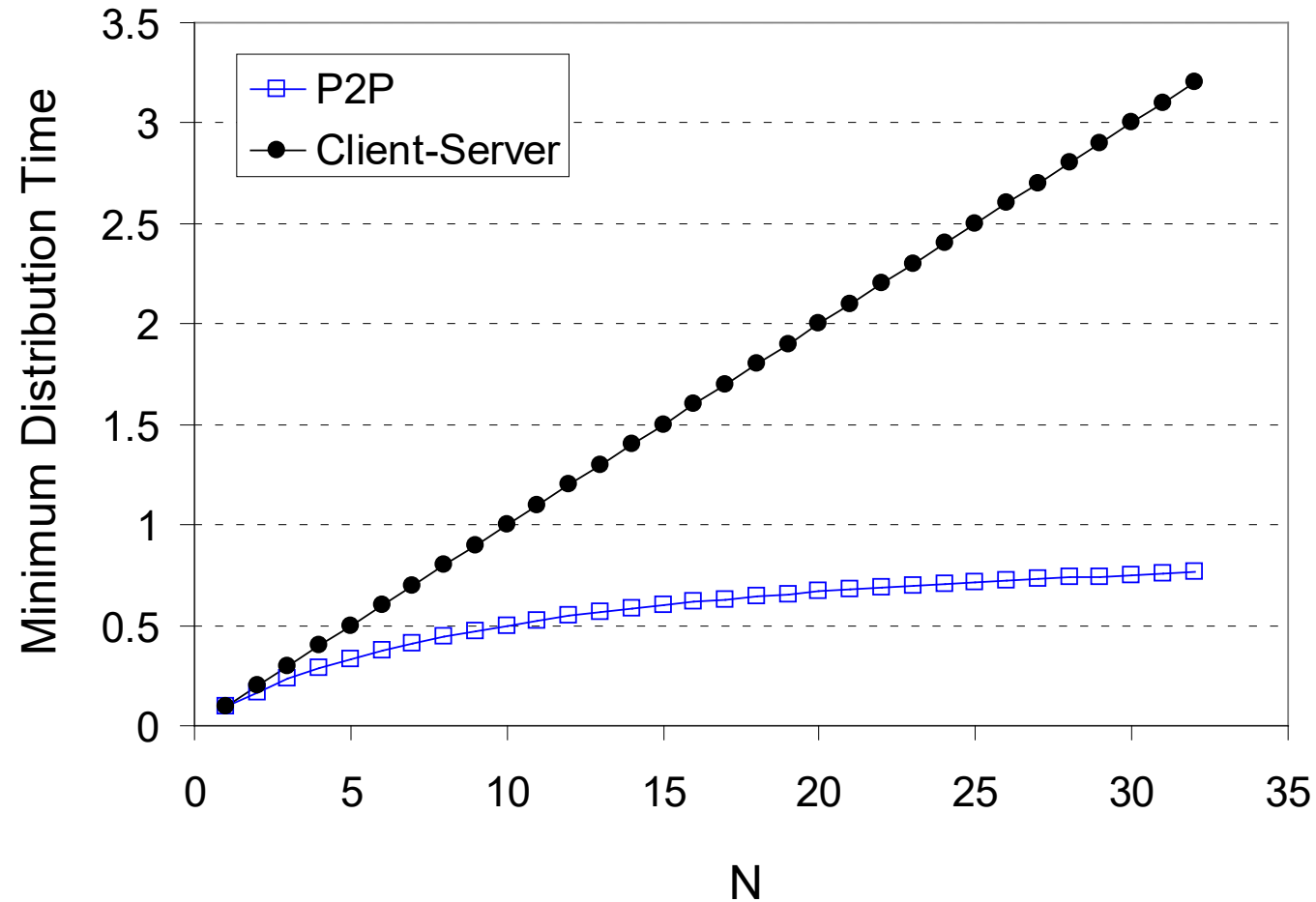 - max upload rate (limiting max download rate) is $u_s + \Sigma u_i$
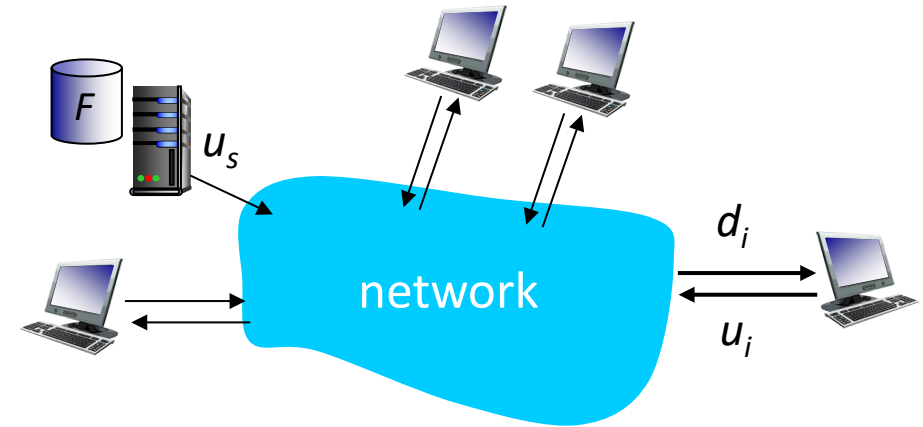
| time to distribute F to N clients using P2P approach | $D_{P2P} \geq max\{F/u_s, F/d_{min}, NF/(u_s + \Sigma u_i)\}$ |

increases linearly in N …

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

Client-server vs. P2P: example

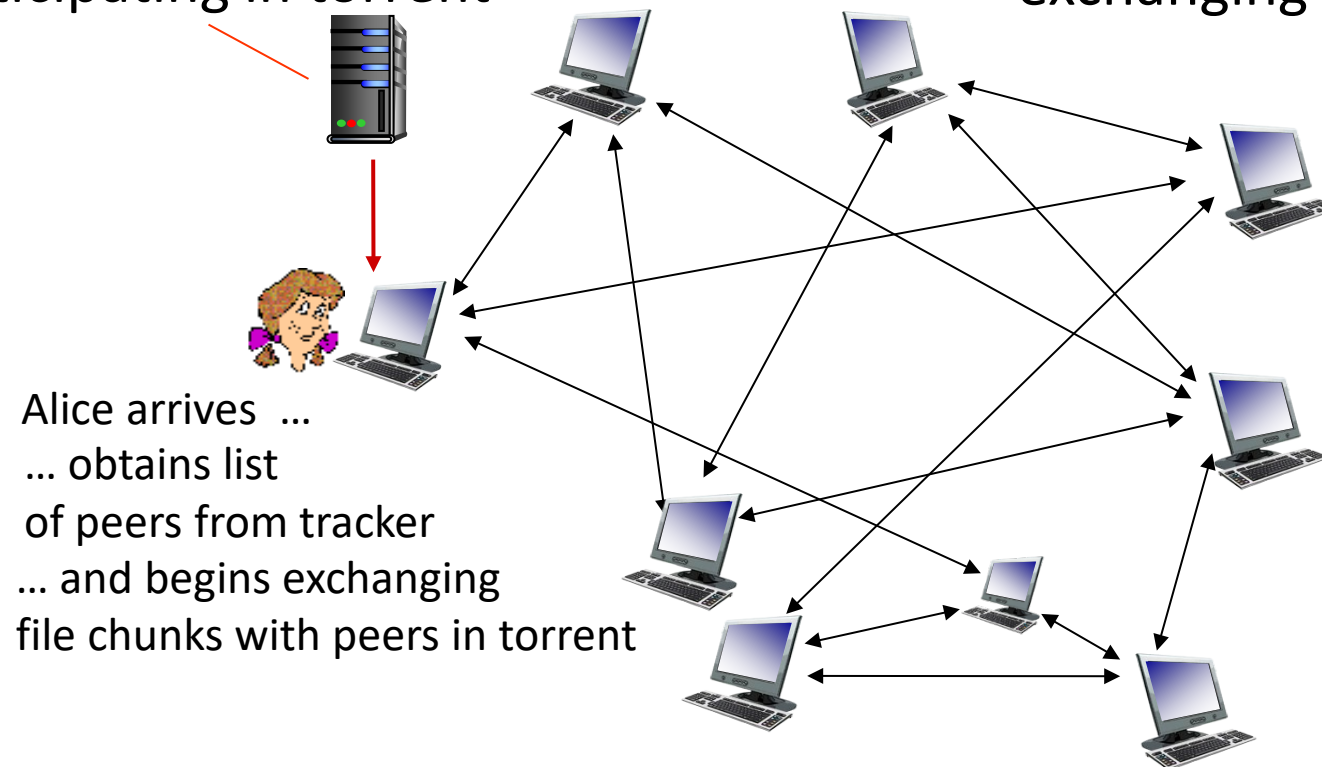client upload rate = u, F/u = 1 hour, $u_s = 10u$, $d_{min} \geq u_s$

P2P file distribution: BitTorrent

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

tracker: tracks peers participating in torrent

torrent: group of peers exchanging chunks of a file

Alice arrives …
… obtains list
of peers from tracker
… and begins exchanging
file chunks with peers in torrent

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**
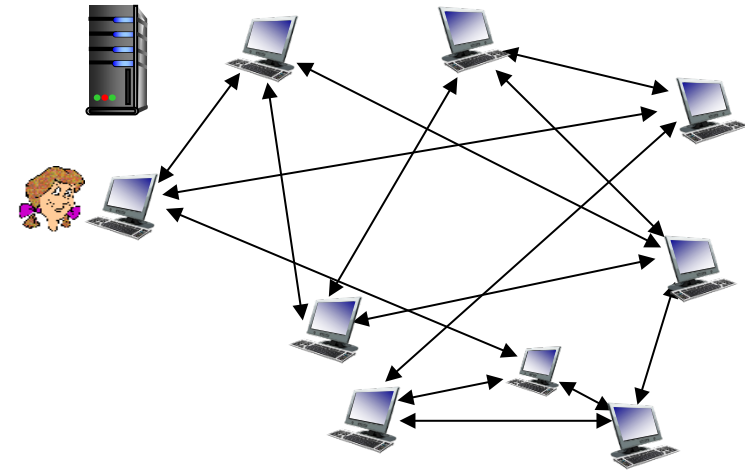
BitTorrent: requesting, sending file chunks

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 !