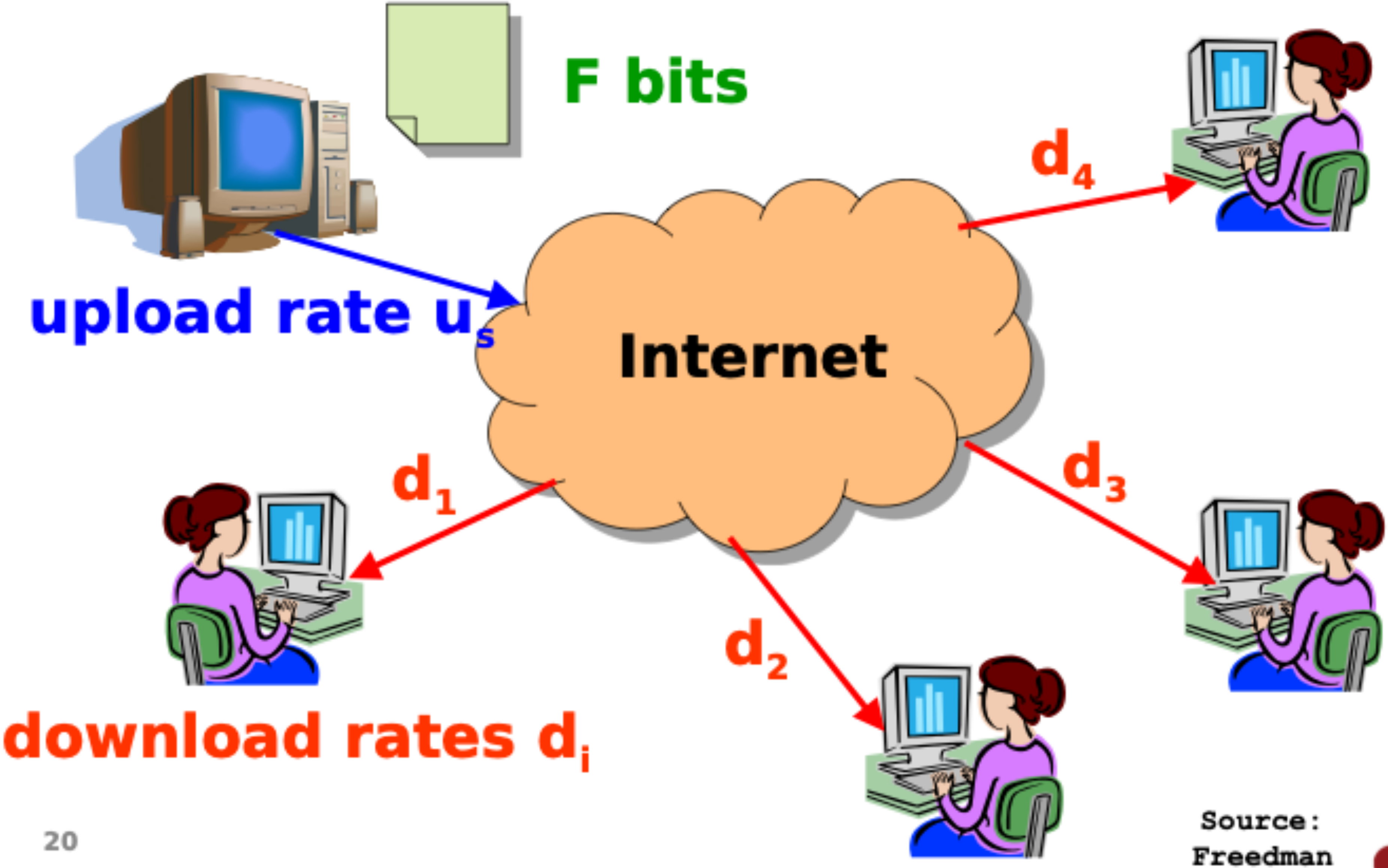
UNIVERSITY OF COPENHAGEN

Application Layer: Peer to Peer File Sharing

Michael Kirkedal Thomsen

Based on slides compiled by Marcos Vaz Salles, adaptions by Vivek Shah and Michael Kirkedal Thomsen

Server Distributing a Large File



Server Distributing a Large File

- Sending an F-bit file to N receivers
  - Transmitting NF bits at rate u_s
  - … takes at least NF/u_s time
- Receiving the data at the slowest receiver
  - Slowest receiver has download rate $d_{min} = min_i\{d_i\}$
  - … takes at least F/d_{min} time
- Download time: $max\{NF/u_s , F/d_{min}\}$

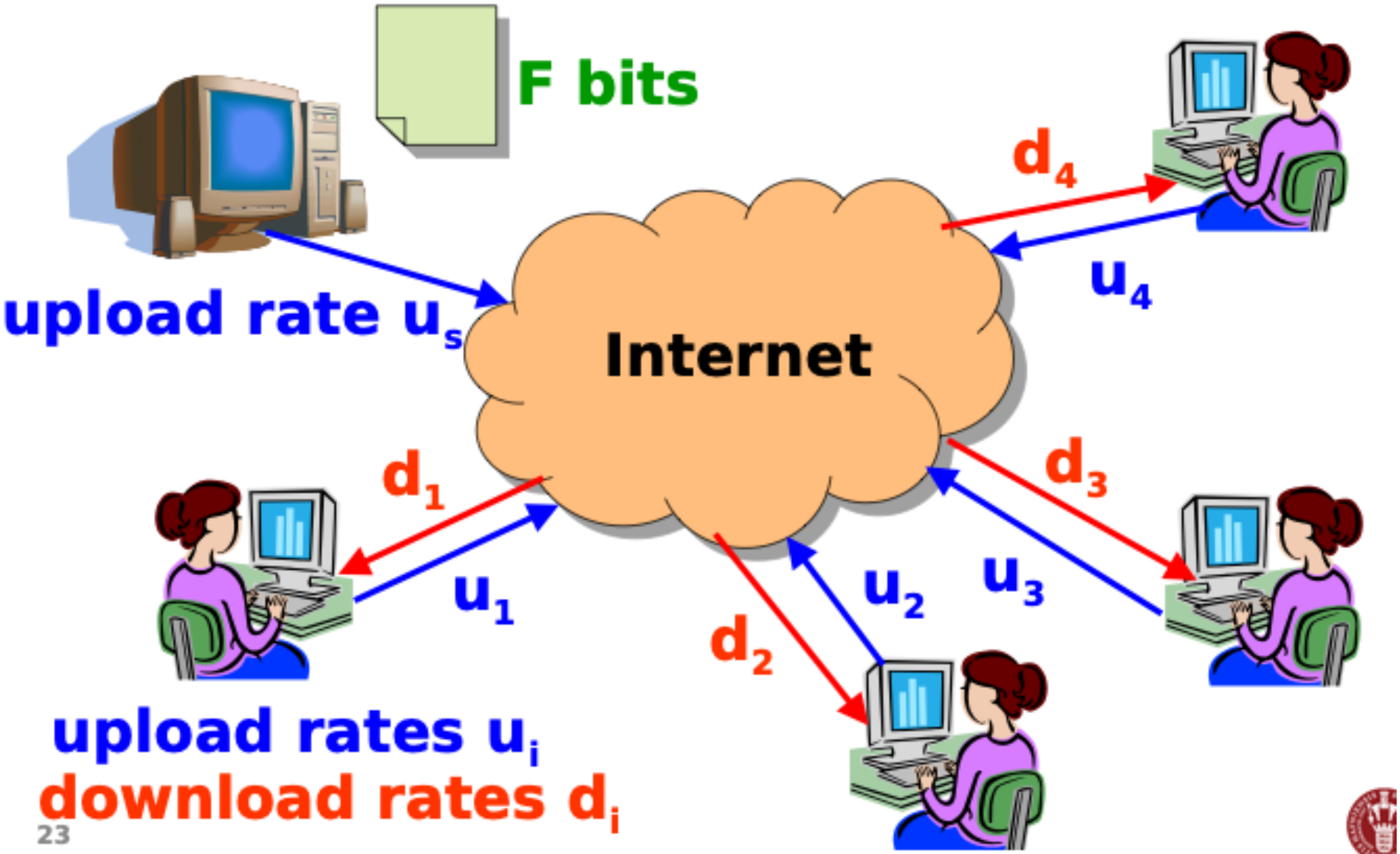
Source:
Freedman

Speeding Up the File Distribution

- Increase the server upload rate
  - Higher link bandwidth at the server
  - Multiple servers, each with their own link
- Alternative: have the receivers help
  - Receivers get a copy of the data
  - … and redistribute to other receivers
  - To reduce the burden on the server

Source: Freedman

Pees Help Distributing a Large File



Peers Help Distributing a Large File

- Components of distribution latency
  - Server must send each bit: min time F/u_s
  - Slowest peer must receive each bit: min time F/d_{min}
- Upload time using all upload resources
  - Total number of bits: *NF*
  - Total upload bandwidth $u_s + sum_i(u_i)$
  - Total: $max\{F/u_s, F/d_{min}, NF/(u_s+sum_i(u_i))\}$
- *Peer to peer is self-scaling*
  - *Download time grows slowly with N*
    - Client-server: $max\{NF/u_s, F/d_{min}\}$
    - Peer-to-peer: $max\{F/u_s, F/d_{min}, NF/(u_s+sum_i(u_i))\}$

Source: Freedman

Peer-to-Peer Networks: BitTorrent

- BitTorrent history
  - 2002: B. Cohen debuted BitTorrent
- Emphasis on efficient fetching, not searching
  - Distribute same file to many peers
  - Single publisher, many downloaders
- Preventing free-loading
  - Incentives for peers to contribute

Source: Freedman

BitTorrent: Tracker

- Infrastructure node
    - Keeps track of peers participating in the torrent
    - Peers register with the tracker when it arrives
- Tracker selects peers for downloading
    - Returns a random set of peer IP addresses
    - So the new peer knows who to contact for data
- Can have "trackerless" system
- Using distributed hash tables (DHTs)

Source: Freedman

BitTorrent: Chunk Request Order

- Which chunks to request?
  - Could download in order
  - Like an HTTP client does
- Problem: many peers have the early chunks
  - Peers have little to share with each other
  - Limiting the scalability of the system
- Problem: eventually nobody has rare chunks
  - E.g., the chunks need the end of the file
  - Limiting the ability to complete a download
- Solutions: random selection and rarest first

Source: Freedman

BitTorrent: Rarest Chunk First

- Which chunks to request first?
  - Chunk with fewest available copies (i.e., rarest chunk)
- Benefits to the peer
  - Avoid starvation when some peers depart
- Benefits to the system
  - Avoid starvation across all peers wanting a file
  - Balance load by equalizing # of copies of chunks

Source: Freedman

Free-Riding in P2P Networks

- Vast majority of users are free-riders
    - Most share no files and answer no queries
    - Others limit # of connections or upload speed
- A few “peers” essentially act as servers
    - A few individuals contributing to the public good
    - Making them hubs that basically act as a server
- BitTorrent prevent free riding
    - Allow the fastest peers to download from you
    - Occasionally let some free loaders download

Source: Freedman

Bit-Torrent: Preventing Free-Riding

- Peer has limited upload bandwidth
  - And must share it among multiple peers
  - Tit-for-tat: favor neighbors uploading at highest rate
- Rewarding the top four neighbors
  - Measure download bit rates from each neighbor
  - Reciprocate by sending to the top four peers
- Optimistic unchoking
  - Randomly try a new neighbor every 30 seconds
  - So new neighbor has a chance to be a better partner
  - Compatible peers find each other

Source: Freedman

Peer-to-Peer Naming

- But…
  - Peers may come and go
  - Peers need to find each other
  - Peers need to be willing to help each other

Source: Freedman

Locating the Relevant Peers

- Three main approaches
  - Central directory (Napster)
  - Query flooding (Gnutella)
  - Hierarchical overlay (Kazaa, modern Gnutella)
- Design goals
  - Scalability
  - Simplicity
  - Robustness
  - Plausible deniability

Source: Freedman

Recap: Streaming multimedia: DASH

- *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: Kurose & Ross

P2P media streaming: Octoshape

- Commercial hosting not cable to home, founded 2003
    - Co-founder Stephen Alstrup
    - Broadcasting fee is paid by broadcasters
    - Free for consumers
- Audio and Video, 32kbps to 800kbps
- Mesh based, bit-torrent like, Content Server pushes content to some peers, it propagates from there
- Peers advertise their joining to everyone (?)
    - Probably to some network or geographic topography, esp. if the live content is popular, this can cause a notification storm. [speculation]
- Eurovision in 2006, 2007, Tour de France in 2007 with ~1.5Mbps High Quality
- Requires a web browser, any streaming client and Octoshape clien

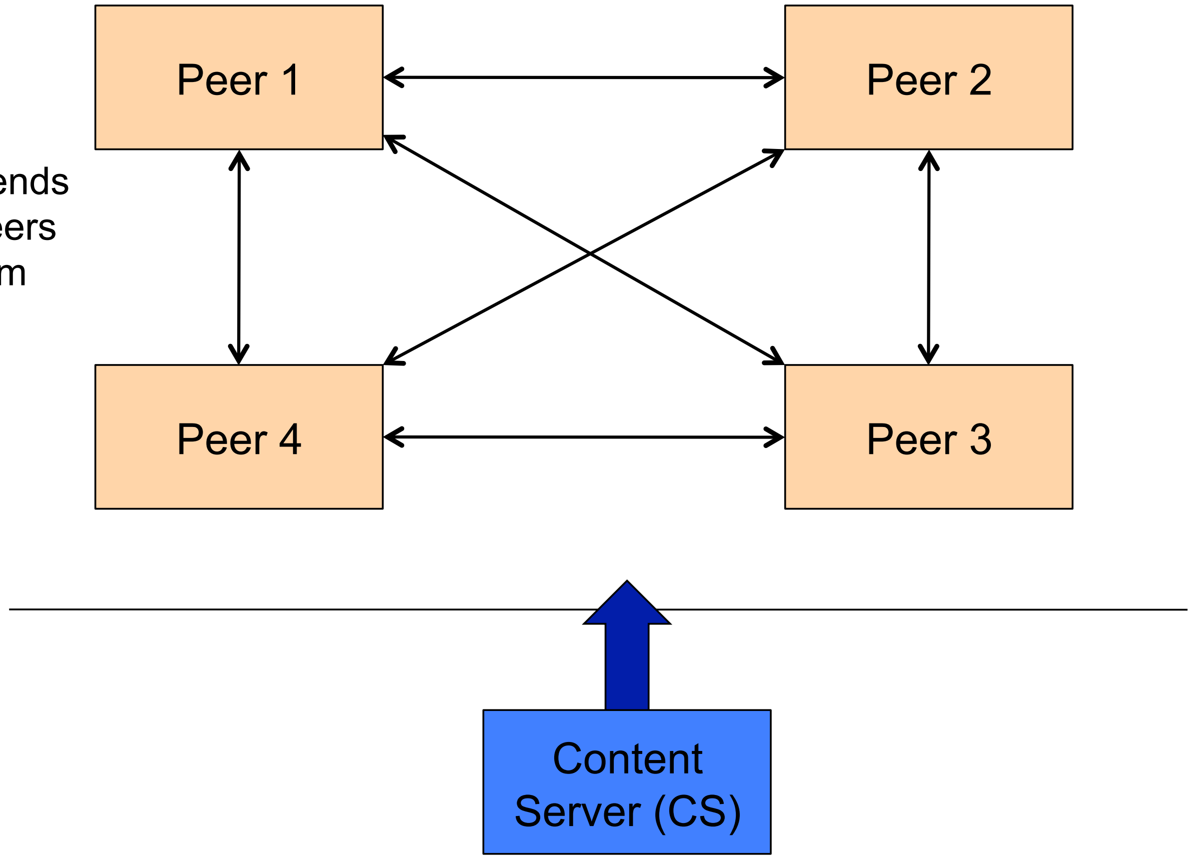
Source: Jörg Ott

P2P media streaming: Octoshape

Peer decides to split the streams in to k-parts.

1) Stand-by peers
2) Connected peers

No of parts depends on number of peers downloading from that peer



Source: Jörg Ott

Summary

- P2P applications
  - Self scalability
  - BitTorrent – Popular P2P file sharing protocol
  - Rarest chunk first, fair trading + optimistic unchoking
  - Many implementation of media streaming
    - Today hybrids via Amazon services