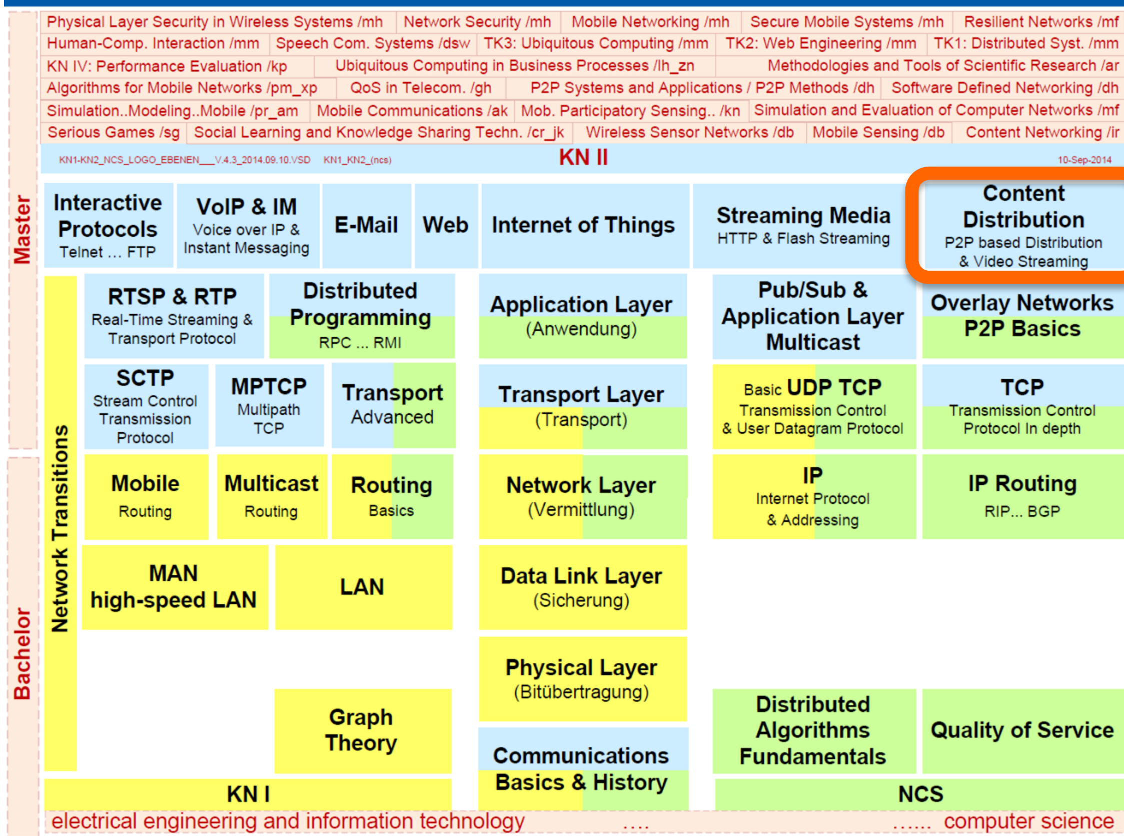
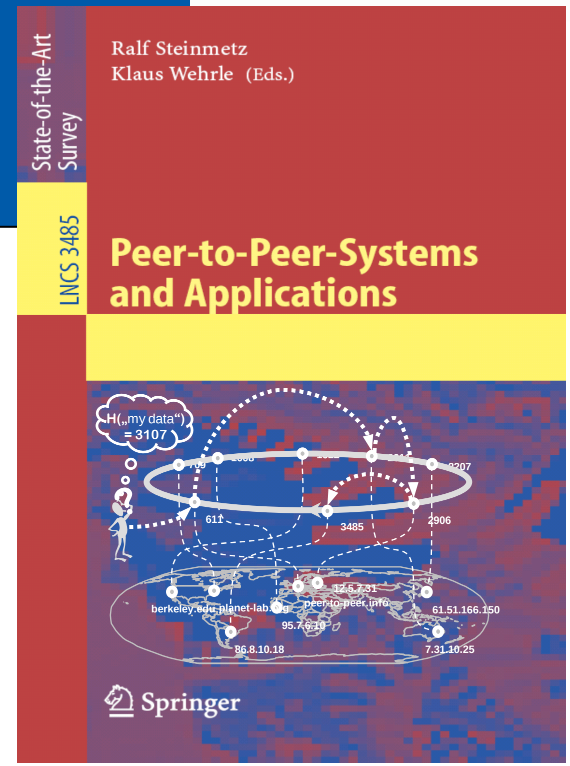
Communication Networks II

Peer-to-Peer Technology
-
Content Distribution and Video Streaming

TECHNISCHE UNIVERSITÄT DARMSTADT





Prof. Dr.-Ing. **Ralf Steinmetz**
KOM - Multimedia Communications Lab

Overview

1 Cooperative File Sharing (like BitTorrent)

Involved nodes / roles:

Tracker

- non-content-sharing node
- actively tracks
  - Swarm: all peers (seeders and leeches)
  - Status of downloaded data - volume
- one swarm per file

Seeders

- have complete copies of the desired content

Leechers

- incomplete copies of the desired content
- leeches try to download missing chunks

1.1 Cooperative File Sharing (like BitTorrent): Mechanisms

Characteristics

- one network per file
- file is split into chunks
  - typically 256kB
- and ... Tit-for-tat..

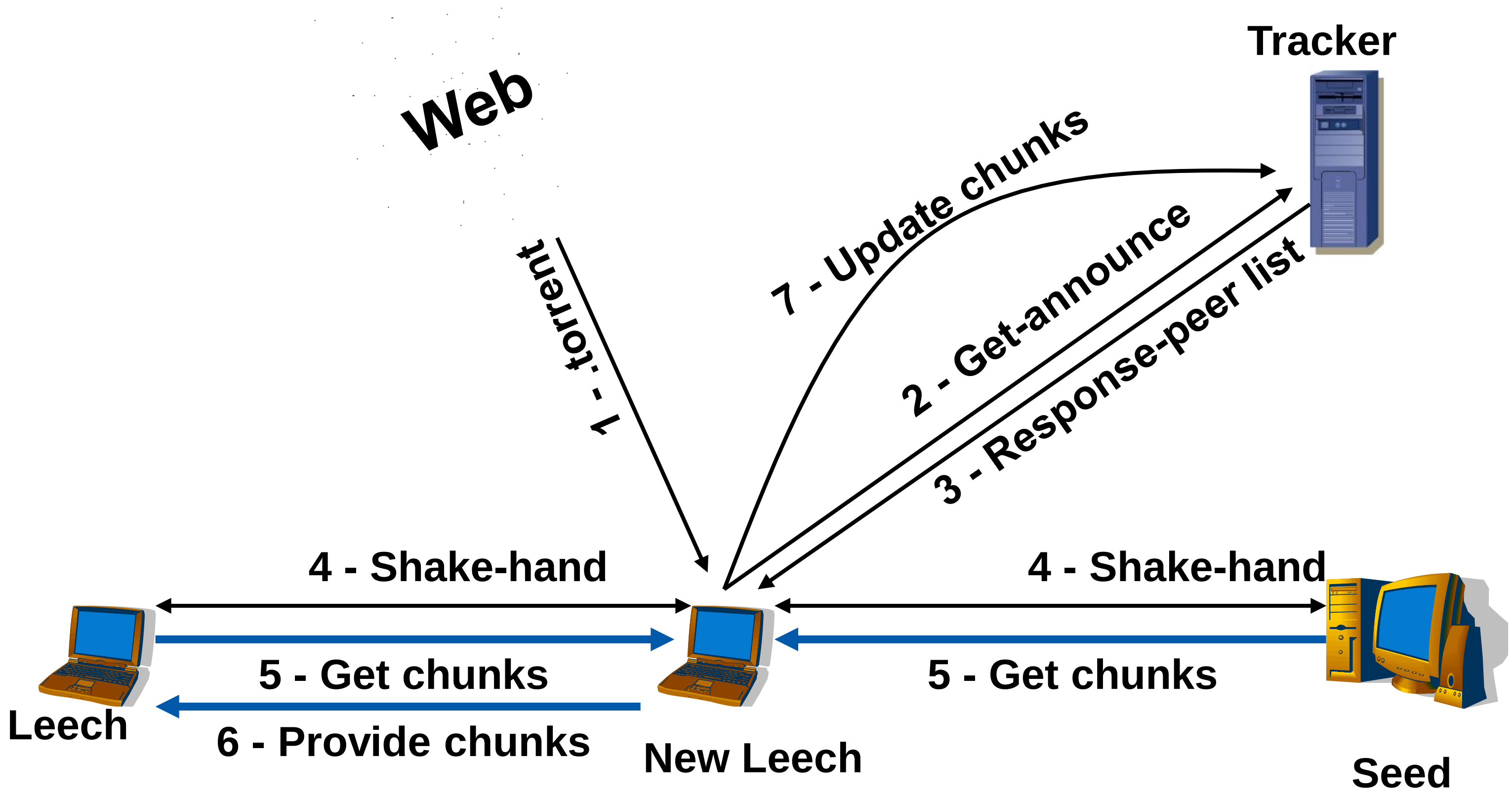
tit-for-tat exchange strategy

- if I give you – you give me
  - being optimistic about unknown nodes
- attempt to reach Pareto efficiency
  - Pareto efficient means:
    - no individual improve his situation without another loosing by at least the same “degree”
  - → no one can get faster download speeds without hurting someone else's download speed

with

- nodes download rarest chunks first
- new nodes download random chunks first

Cooperative File Sharing (like BitTorrent): A Scenario



Phases of BitTorrent

1. Get torrent from Web

- Several websites offer torrents
- Torrent contains information about the file:
  - filename, size, checksum, and a corresponding tracker

2. Get announce, contact tracker

- Tracker maintains a list of active peers sharing the file
- One tracker per file,
  one network (swarm) per file

3. Response peer list

- Peer list contains up to 50 peers
  - Seeders and leechers
- Large number of peers sharing the same file
  → allows for parallel downloads of chunks

4. Shake-hand

- Peer establishes connection to 20-40 peers from the peer-list
- Checks periodically their bandwidth capabilities
  → prioritized list
- Uses just 4 for exchanging chunks
- Various replacement strategies

5. Get chunk

- Peers request chunks
  - using the rarest first policy
- Parallel chunk download allows for faster file downloads

6. Provide chunk

- Tit-for-tat strategy,
  - peers give as long as they receive some chunks
- Slow startup phase, as not chunks provided

7. Update chunk information at tracker

1.2 Cooperative File Sharing (like BitTorrent): Evaluation

Strengths

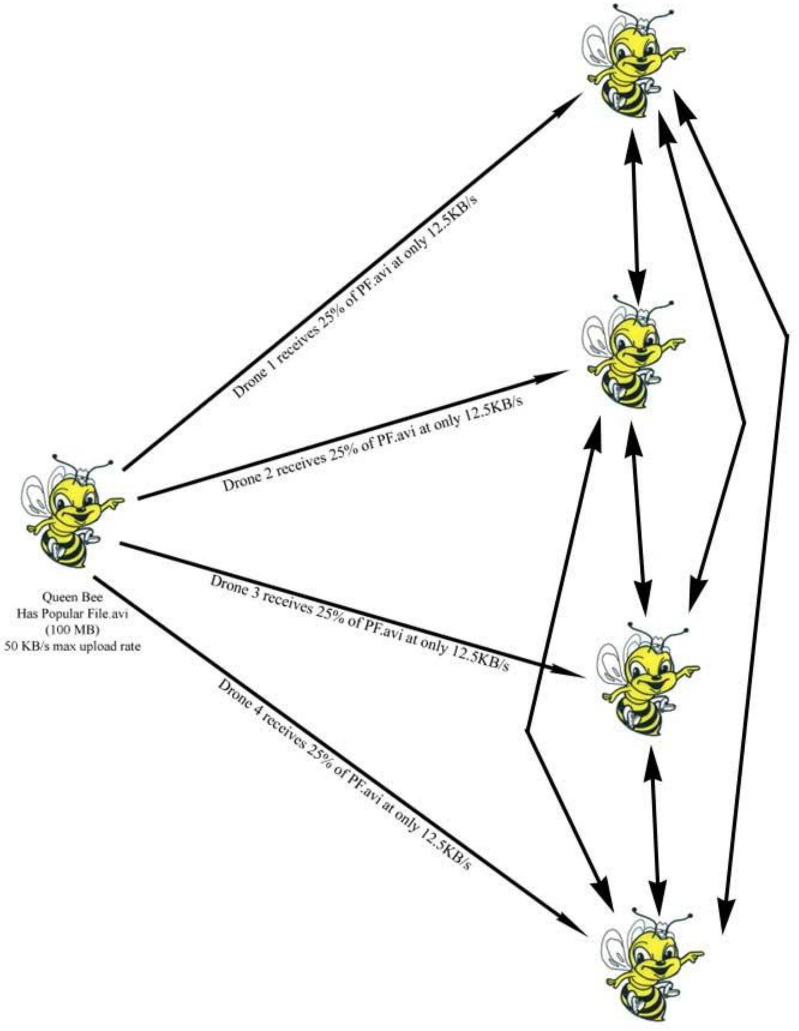
- Good bandwidth utilization
- Limit free riding
  - tit-for-tat
- Limit leech attack
  - coupling upload & download
- Spurious (“unechte”) files not propagated
- Open source implementations
- Preferred selection
  - for legal content distribution

Drawbacks

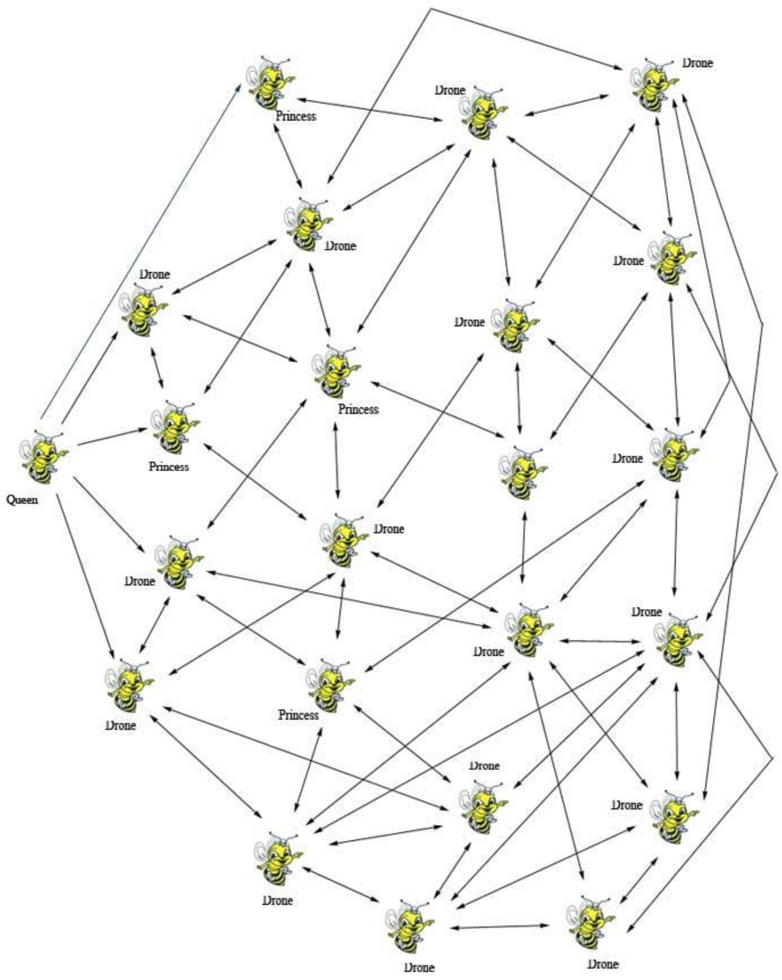
- Small files
  - lead to latency, overhead
- Random list of peers
  - naïve
- Central tracker server needed to bootstrap swarm
  - Single point of failure
  - Scalability issue
- Robustness
  - System progress dependent on altruistic nature of seeds (and peers)
- Cannot totally avoid malicious attacks at leeches
- Centralized supervising tracker – trust?

Cooperative File Sharing (like BitTorrent): Evaluation

ideal BitTorrent operation



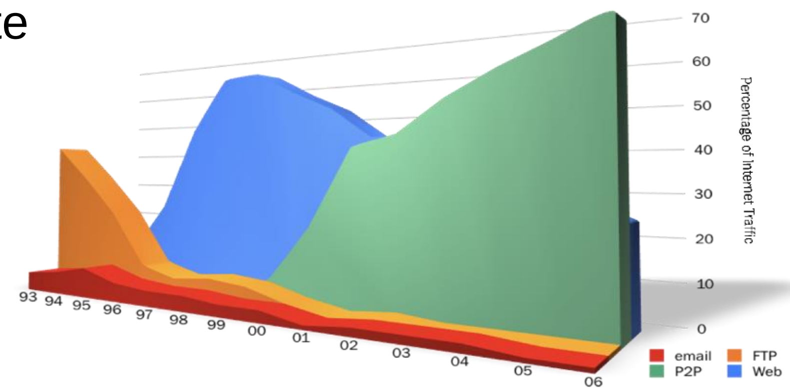
BitTorrent at a certain point in time



TECHNISCHE UNIVERSITÄT DARMSTADT

Video distribution becomes popular in the current Internet

- YouTube:
  - >2 billion views a day by September 2010
  - 24 hours of video uploaded every minute



Source: CacheLogic Research

High costs

- E.g. YouTube 10$/Mbps
- Results in 1 Mio $ per day
- Flash crowds make content unavailable

Problems for ISPs

- Tit-for-tat strategy optimizes for low download times
- Typically chunk provider is far away
- Generates inter-ISP traffic → expensive
- Local providers are more ISP-friendly

Strategies to reduce ISP costs

TECHNISCHE UNIVERSITÄT DARMSTADT

Filter P2P traffic → provide worse quality

Usage-based billing
- Introduce fair flat rates
- volume based flat rates

Caching of P2P content by ISPs

Cooperate of ISPs with P2P application providers
- Building P2P applications aware of the network locality
- P4P: Provider Portal for Applications, http://openp4p.net
- Idea: ISP intervene in step 3
  - Peer requests peer list
  - Tracker contacts P4P server to localize peer list
  - Localized peer list is delivered

Result:
- Chunk providers are in same / near ISP network
- More P2P traffic remains in the ISP network
- Lower costs for ISPs

Global content delivery on the Internet – observation:

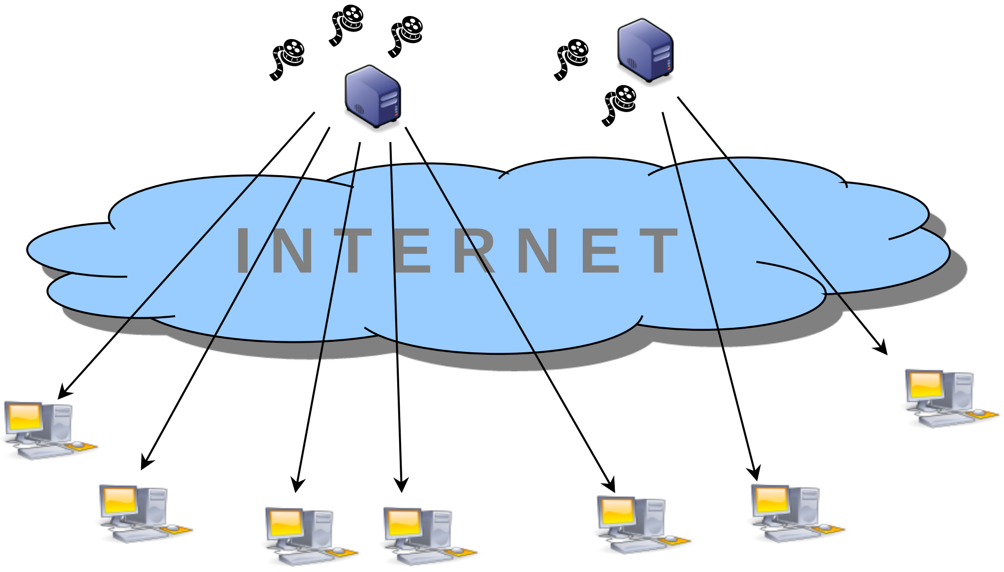
- Increasing number of PC users
- Increasing diversity of Internet devices

All using Internet services

- Not just web pages, pictures but …
- Software (updates, patches)
- Video content

Results in

- →Increasing Internet traffic
- →Increasing number of servers
- Increasing costs
- More bottlenecks
- More download failures



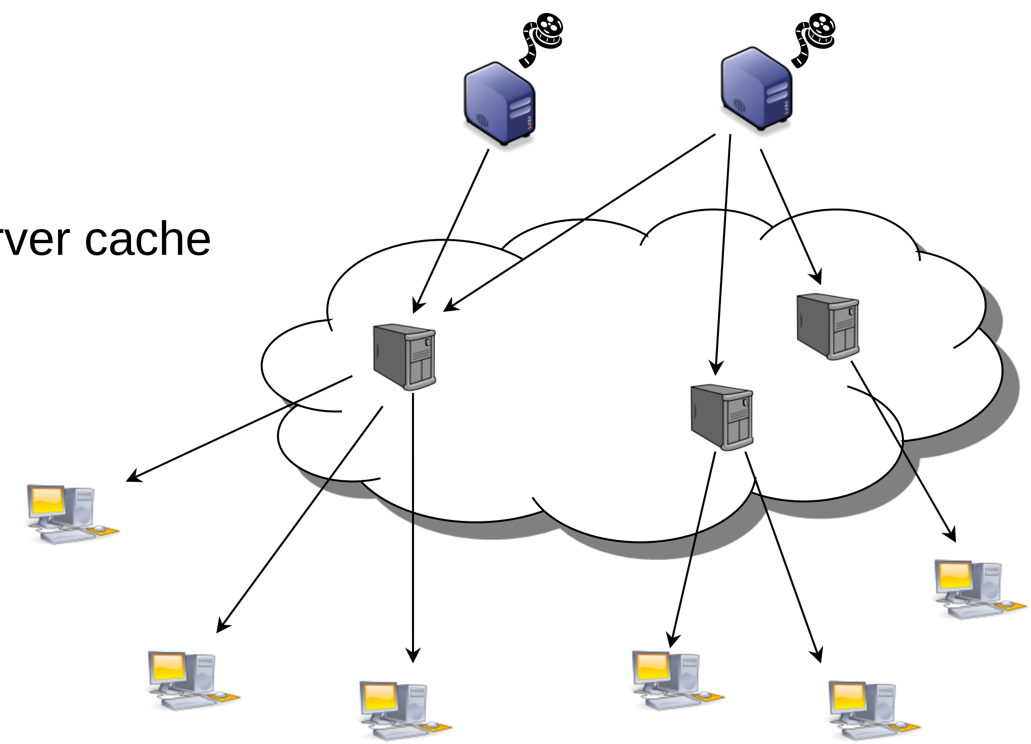
Traditional Content Delivery Networks (CDNs)

Managed network of servers

- Distributed across the Internet
- Host content on demand for paying customers (content providers)
- Aka: Content Distribution Networks

How it works?

- User request is redirected to a close CDN server
- The content is served from server cache
- Server selection is based on
  - User location
  - Network conditions



CDNs – Discussion of Examples

Akamai

- Single servers distributed around the world

Limelight

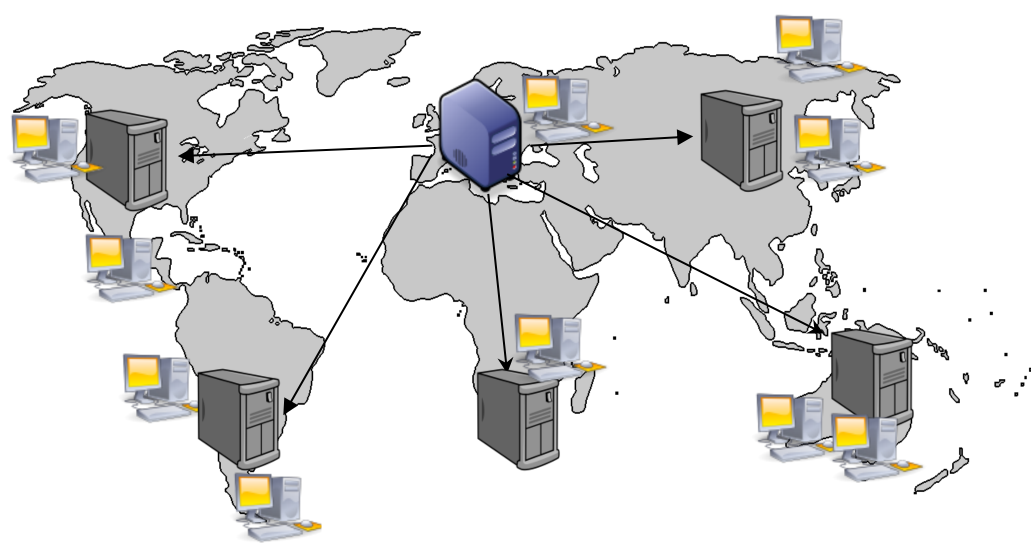
- Few data centers at strategic positions

Benefits

- Faster response time (for cached content)
- Less transit traffic
- Better load balancing / scalability

Drawbacks

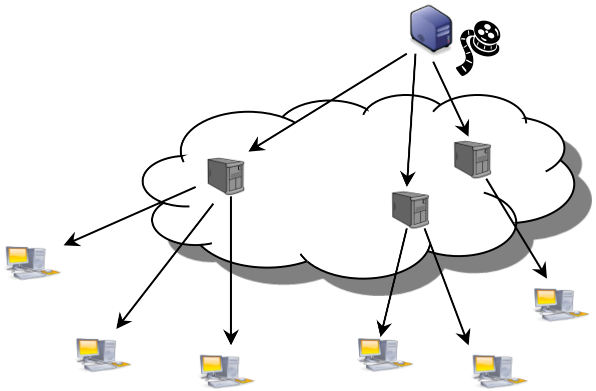
- High costs
  - Server maintenance
  - Bandwidth costs
- Still problems with flash crowds
  - If too many users request the same content simultaneously



3.1 Comparison CDN and P2P

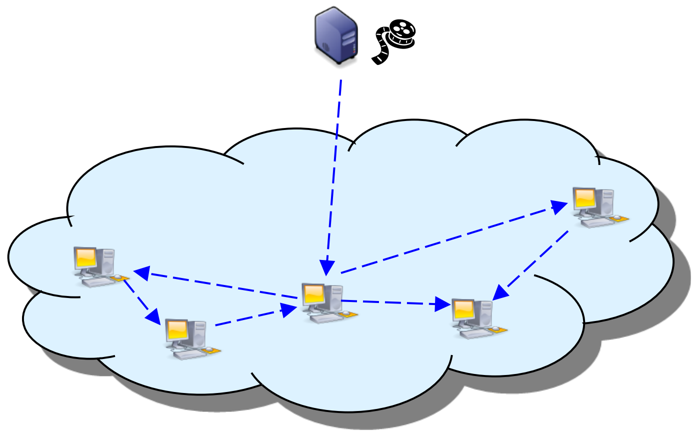
P2P:

- No infrastructure cost
- Supply grows linearly with demand
- Simple distributed, randomized algorithms
- No QoS



CDN:

- Initial infrastructure cost
- Centralized scheduling algorithms
  - → Load-balancing
- Network efficiency
  - → Reduced inter-domain traffic
- Capable of supporting QoS
  - → Predictable download times



Comparison CDN and P2P

Can they be combined?

Use P2P to complement CDN

- P2P reduces load on the CDN, covers areas where CDN is not installed
- Must be able to control, or "shape", P2P traffic

→ Peer-assisted CDN

Use CDN to complement P2P

- CDN steps in when peer-based distribution is falling short, enabling QoS
- Must be able to detect when peers do not meet the delivery time guarantee

→ P2P-CDN:
P2P with CDN as backup

3.2 Peer-assisted CDN

Idea:

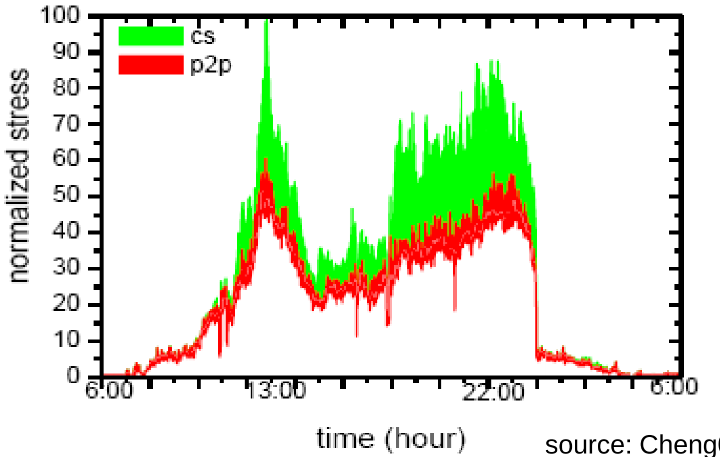
- Offload CDN servers
- Let old users upload the content to new users

Analysis:

- Works well for popular content
- But: Asymmetric user bandwidth (e.g. for DSL 1:8)
  → CDN server has to fill the bandwidth gap
- Cannot be used transparently with protocols, such as HTTP, FTP
  →Requires other protocols and special clients
  - (cf. simple downloads via web browser)

Examples:

- Patches in Blizzard's World of Warcraft
- BBC iPlayer
- Many others: CacheLogic, Pando Networks, Redswoosh



source: Cheng07

P2P-CDN: Problem Statement

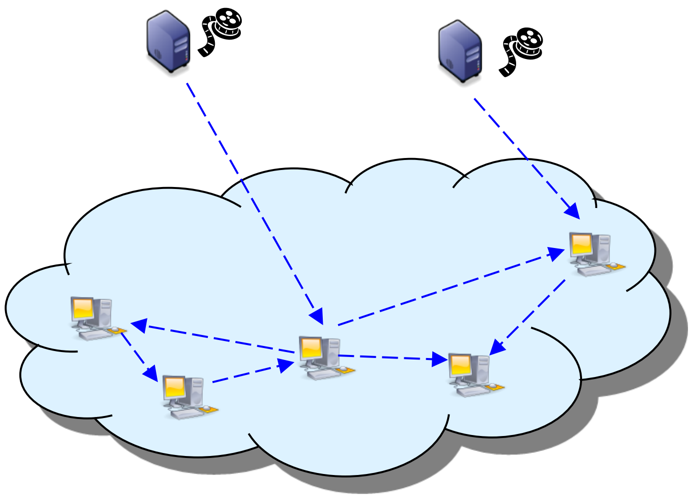
Can we shift the CDN layer to peers?

- Load each content only once from servers
- Serve later requests from peers

Results in a P2P with CDN as backup

Requirements

- Fast downloads
- High overlay hit rate
- Low distribution costs
- Scalable regarding
  - content amount and
  - user number



How can this be done in a P2P environment?

- No central control
- No global information
- Voluntary participation and resource contribution

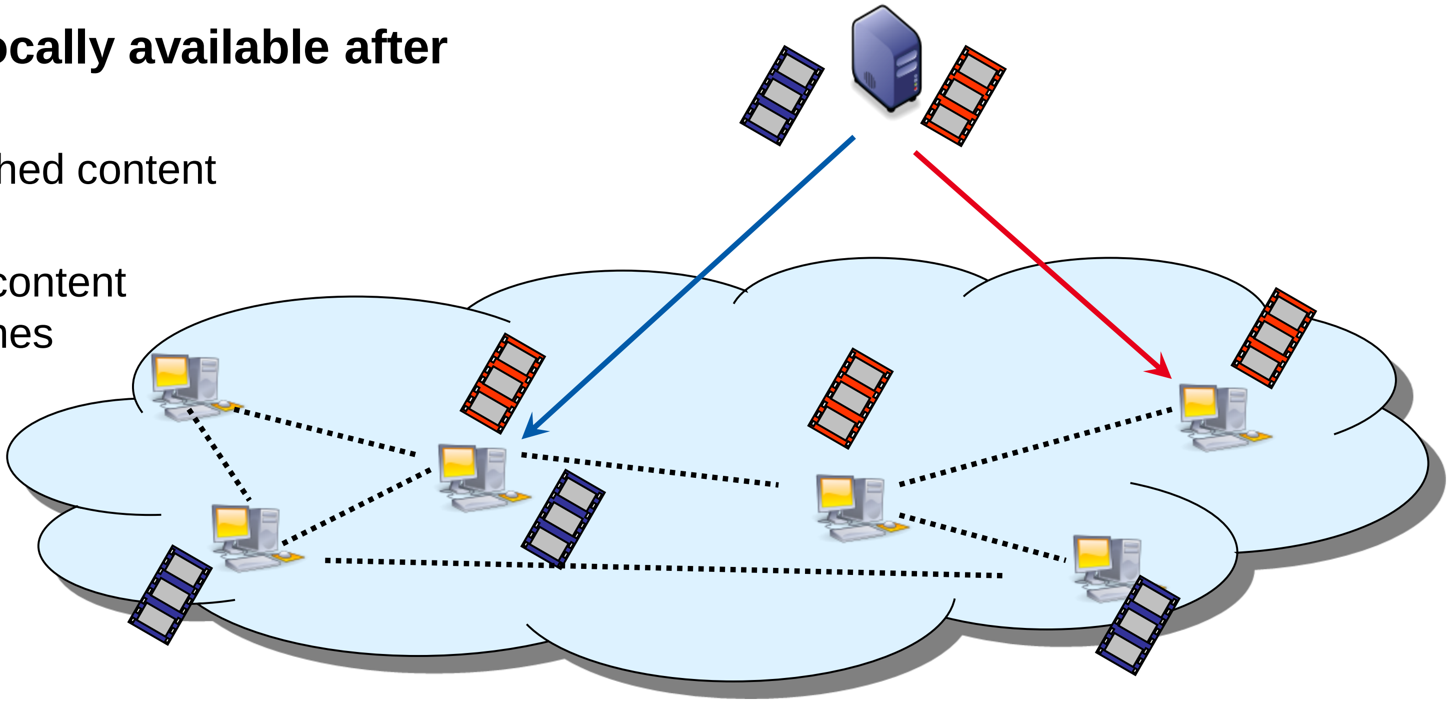
P2P Caching Overlay for Content Delivery

Files are initially stored on content provider servers

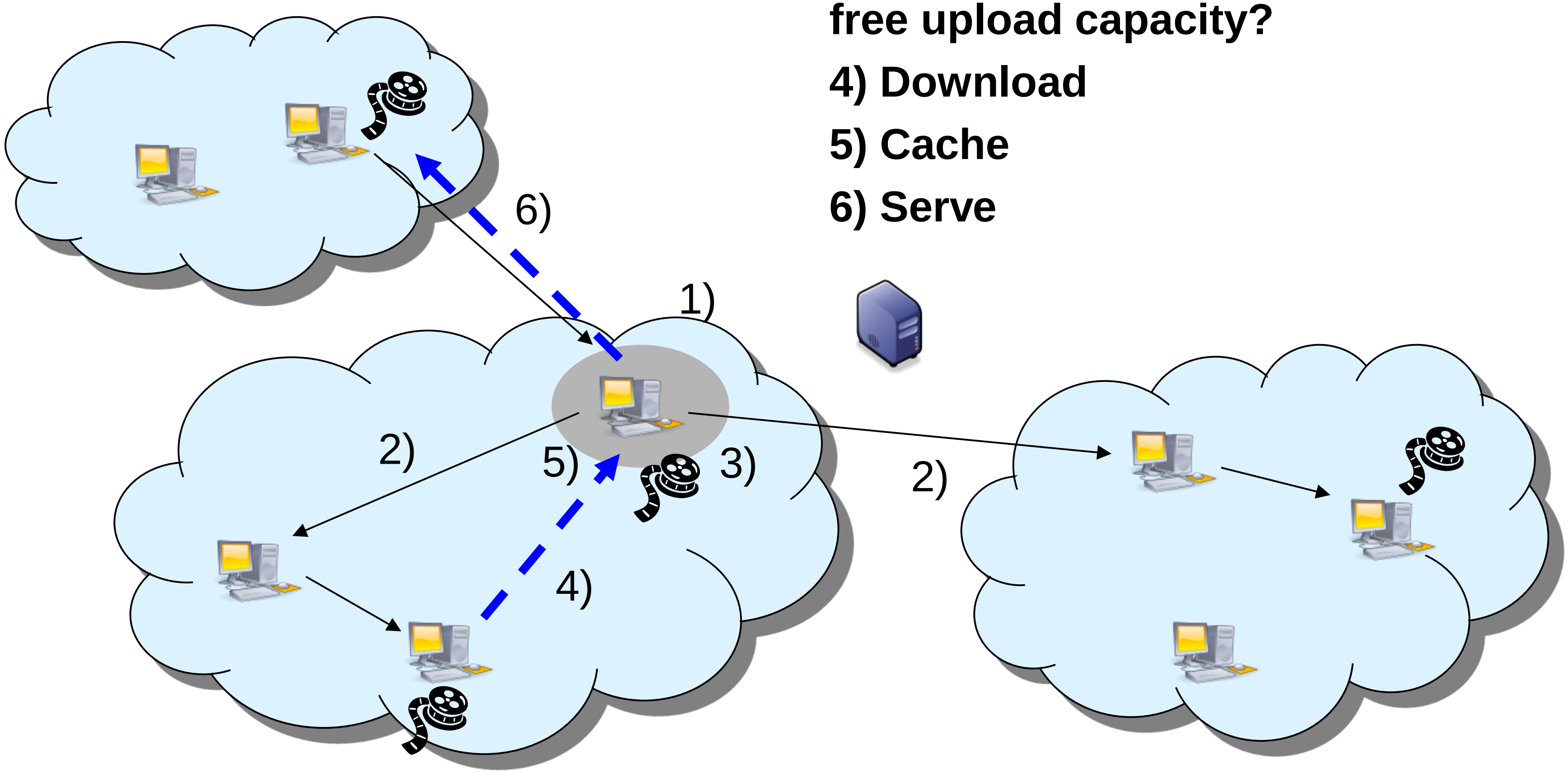
- Downloaded files are stored in peer caches
- Always try to download content from other peer caches
  - Only otherwise from server

Keep content locally available after download

- Find close cached content
- Decide which content to keep in caches



Collaborative Caching Overlay



Procedure

1) Available?

2) Find file: list of source

3) Source selection: any, same AS, free upload capacity?

4) Download

5) Cache

6) Serve

Impact of Seeding Behavior

Peer that finishes the download can

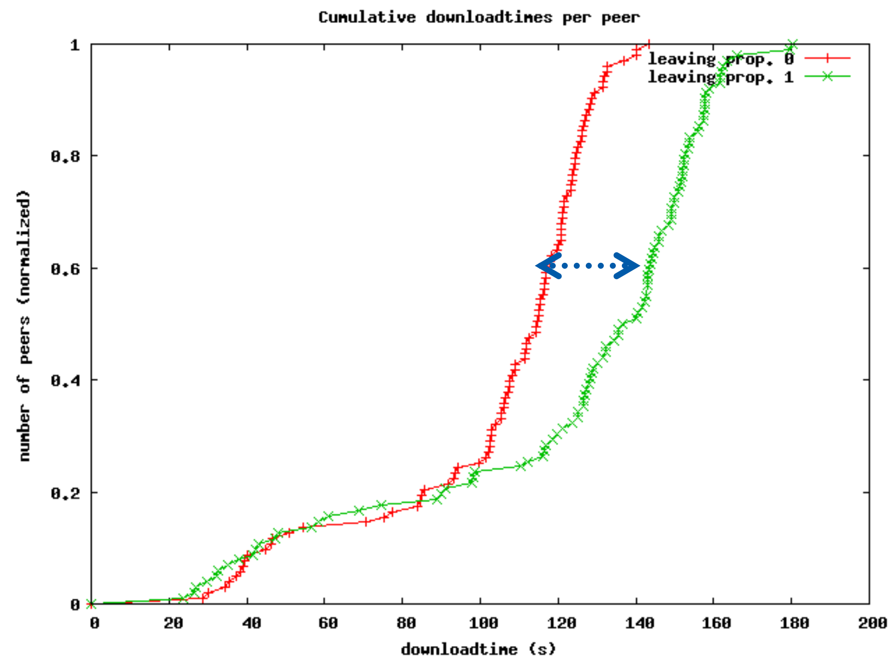
- Leave the network immediately
- or
- Stay online and continue to seed

More seeders result in:

- Shorter download times in second case

Caches can act as additional seeds

- Which swarm to join?
- How much to contribute?
- Fulfill QoS requirements
  - Download rate, delay



Here: 25% faster downloads if finished peers stay online

TECHNISCHE UNIVERSITÄT DARMSTADT

P2P video streaming has become

- increasingly popular approach for streaming (live) content
- many receivers, similar to IPTV

Two approaches for video streaming

- Tree-based overlay
  - Push content delivery
  - Single or multiple tree
- Mesh-shaped overlay
  - Pull content delivery (swarming)
  - Like BitTorrent with modified chunk selection

Research issues

- Streaming topology
- Scalable video codecs (more bandwidth, better quality)

Existing Applications

Mesh-based P2P streaming applications

- Joost
- Octoshape
- PPLive
- Zattoo
- PPStream
- SopCast
- TVants

Tree-based P2P streaming applications

- PeerCast
- Conviva

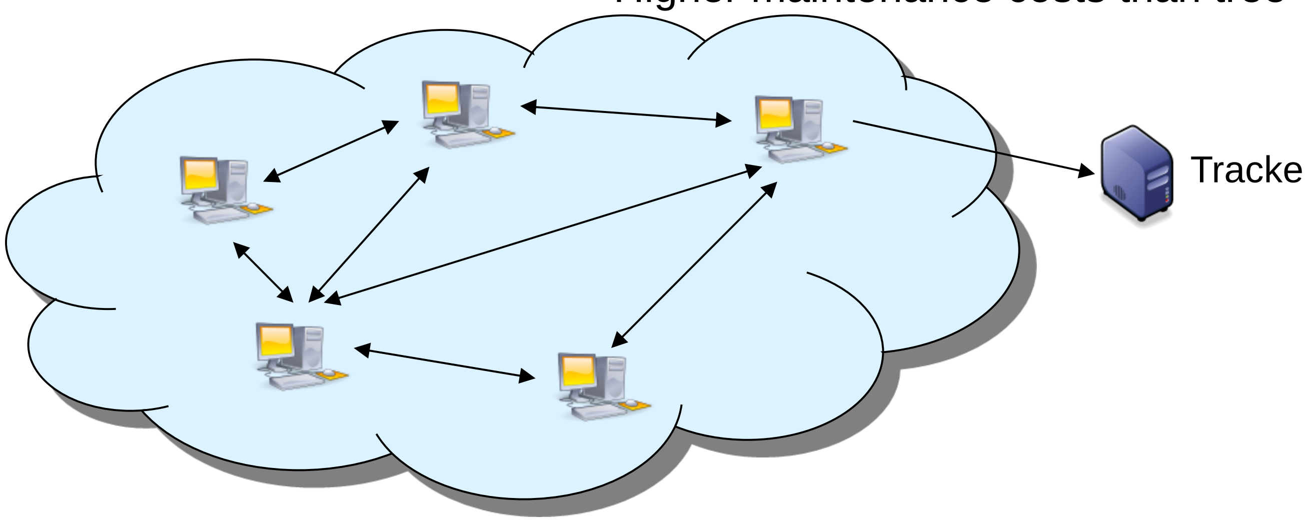
4.1 Mesh-based Video Streaming

Idea
- Similar to BitTorrent
- Peers download the content from each other

But:
- Content is pulled from the network
- Uses modified chunk selection
  - Higher priority for chunks that are about to be played.

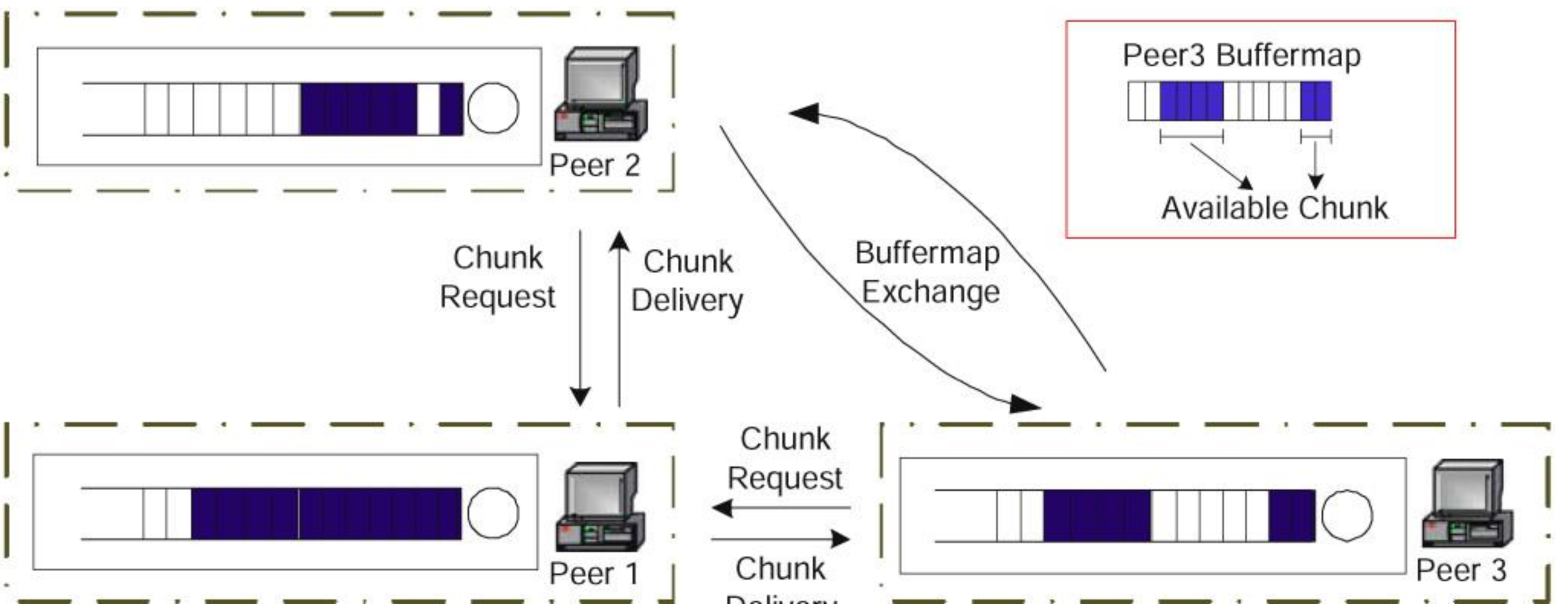
Pros
- More resilient to churn
- Upload requirements flexible

Cons
- Redundant chunks
- Lower efficiency than tree-based topology
- High delay, due to exchange of buffer maps.
- Higher maintenance costs than tree



Data Pull

- Buffer map is a map of all available chunks at a certain peer.
- Exchanged periodically to announce available chunks.



Mesh-based Streaming Protocol

1. Step

- A new peer registers at the tracker
  to join the group watching the same channel

2. Step

- Tracker returns
  an initial peer list to a new peer

3. Step

- Peers within same channel
  exchange peer lists
  - using gossiping and aggregate the peer lists

4. Step

- A new peer randomly selects peers from the list and exchanges information (e.g.: peer status, buffer map

5. Step

- New peer decides about order and priority of data chunks to be requested

6. Step

- Peer requests data from connected peers

Joost

Internet TV service developed by Niklas Zennström and Janus Friis (founders of Skype and Kazaa)

P2P based until 2008, now C/S with a Flash-based web player

- Legal issues due to copyright protected media content

Mesh-based P2P Video-on-Demand Streaming Application

Videos were split up into chunks of 5-10 seconds of the video

Chunks were encrypted

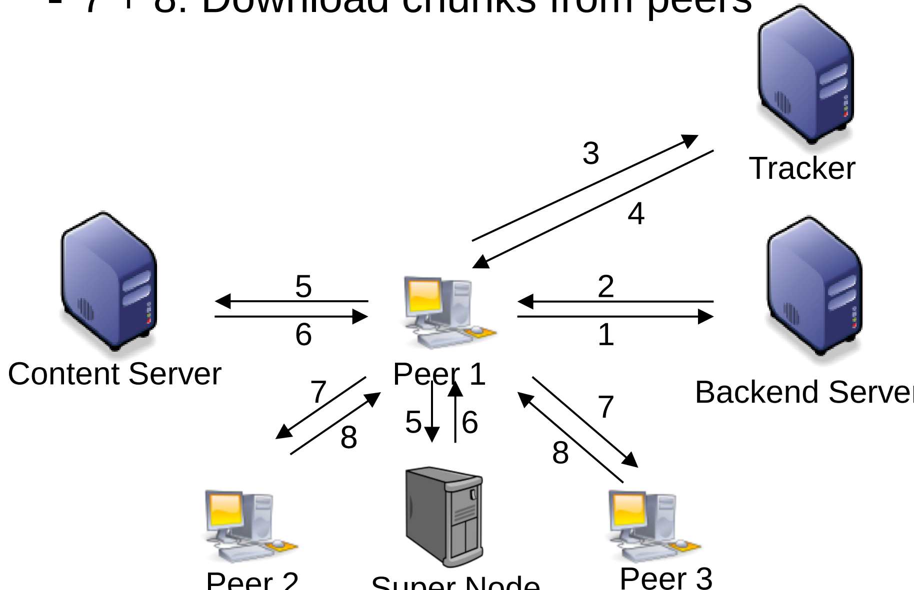
Joost Protocol

Three key components

- Servers
  - Tracker, Version Server,
    Backend Server, Content Server
    - Initial content, Provide channel lists, software updates, addresses of super nodes
- Super nodes
  - Enable channel switching
  - Provide peers addresses within particular channel
- Peers
  - Interconnected in a mesh

Protocol steps

- 1+ 2: Connect to backend server and retrieve channel list
- 3 + 4: Connect to tracker to get IP address of super nodes and content server
- 5 + 6: connect to super nodes to get list of peers providing required chunks
  - Super nodes may redirect peers to content server
- 7 + 8: Download chunks from peers



Octoshape

Commercial P2P Live Streaming Platform

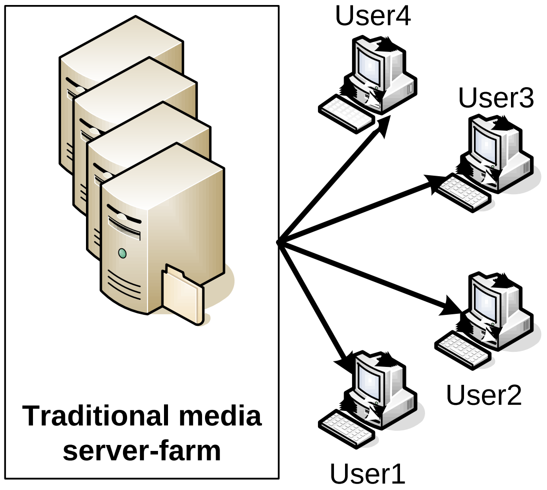
Peers receiving fragments of live stream from other peers

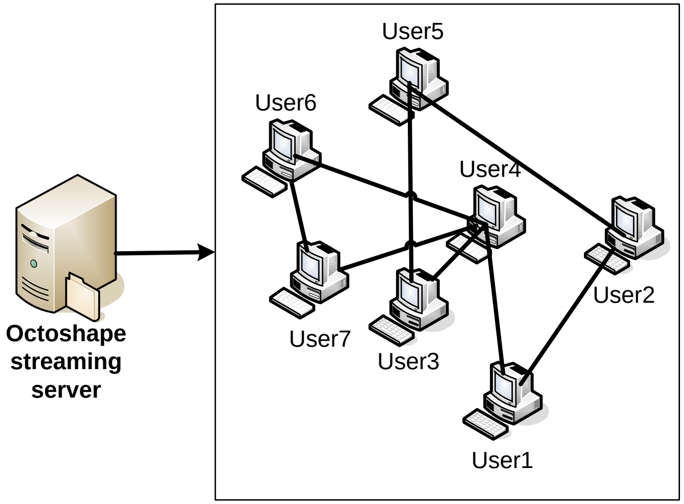
- Client/server approach not cost effective for large scale, high bit-rate live streaming

Results from ESC 2006

- 140.000 viewers from 140 countries
- High quality stream (700kbps)
- Claim: Traffic savings of 97% compared to C/S system

→ Our TUD KOM model: 53% – 79% traffic savings





Octoshape Protocol

Maintains at mesh overlay topology similar to BitTorrent

No tracker server used within the streaming protocol

Peers obtain data from content servers and peers

Each peer maintains

- an address book of other peers within a channel
  - Updated each time a new peer joins a channel
- a list of standby peers
  - which take over if the sender of a stream stops sending

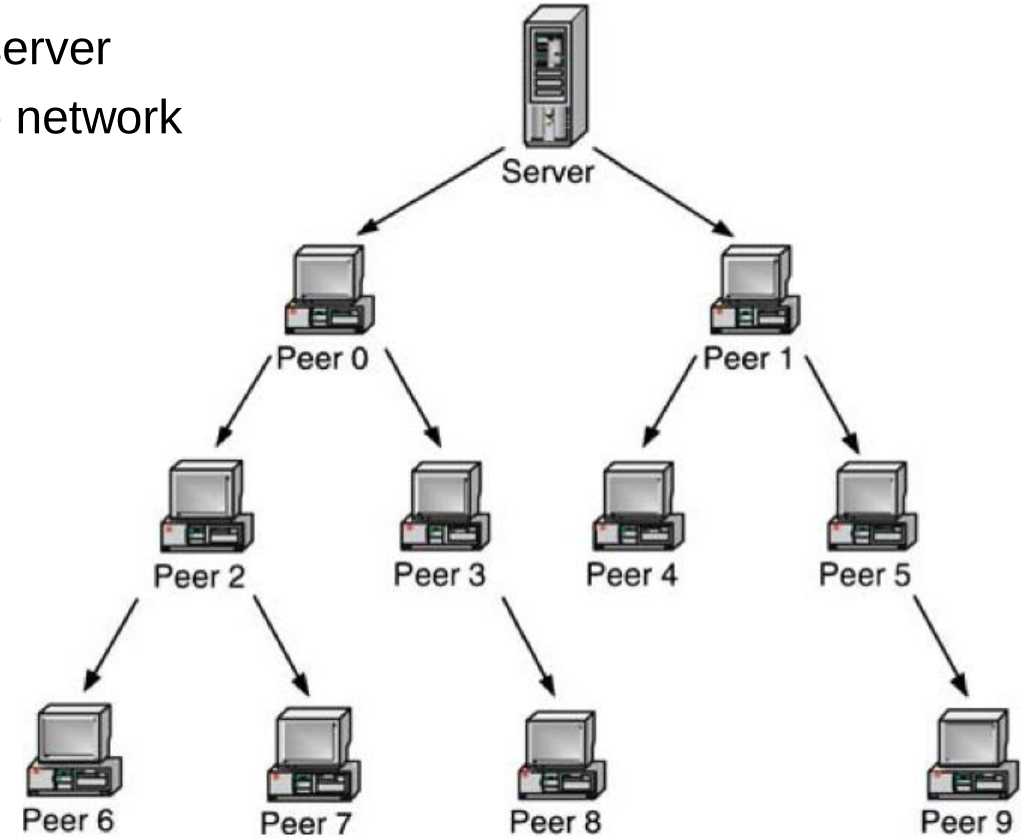
A live stream is

- first sent to a small set of peers within a channel and then
- being spread to the rest of peers in a channel

4.2 Tree-based Video Streaming

Idea:

- Each peer receives the video stream from his parent
- Each peer forwards the stream to its children
- Initial content streamed from server
- Content is pushed through the network



Pros:

- High efficiency
- Low delay

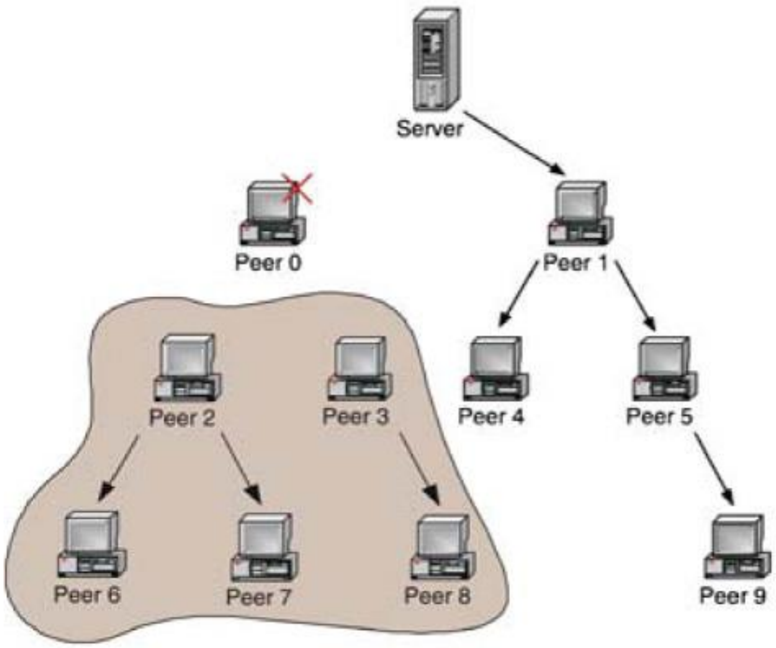
Cons:

- Not very resilient to churn
- Upload requirements high

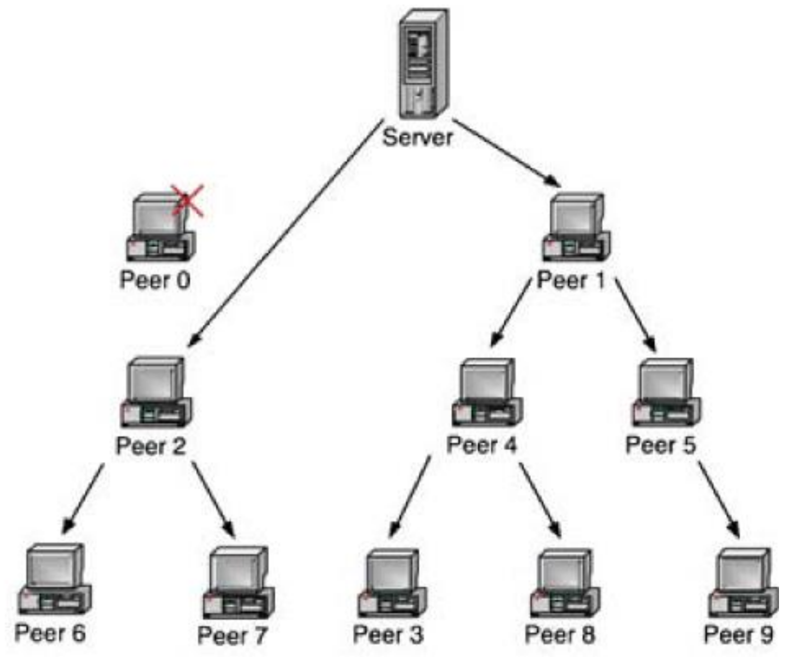
Impact of Peer Churn on Tree based Structure

Churn destroys tree structure

- Mechanisms for robustness needed
- Can be complex and tricky (see Globase.KOM)



a impact of peer churn



b tree recovery after churn

Tree-based Streaming Protocol

1. **A new Peer contacts the tracker server to get list of channel broadcast servers**

2. **Peer contacts broadcast server to join the channel**

3. **Broadcast server either:**
   - Starts streaming content in case of enough idle streaming capacity OR
   - Replies with a list of children to serve the peers requests

4. **In case that the peer is not served by broadcast server**
   - Peer stores the list and
   - Contacts each peer in the list until it finds a node that can serve it

5. **The data stream gets pushed down the tree**

6. **Each peer periodically sends his status to his parent peer**

Peercast

Founded in 2002

peercast.org
p2p broadcasting for everyone

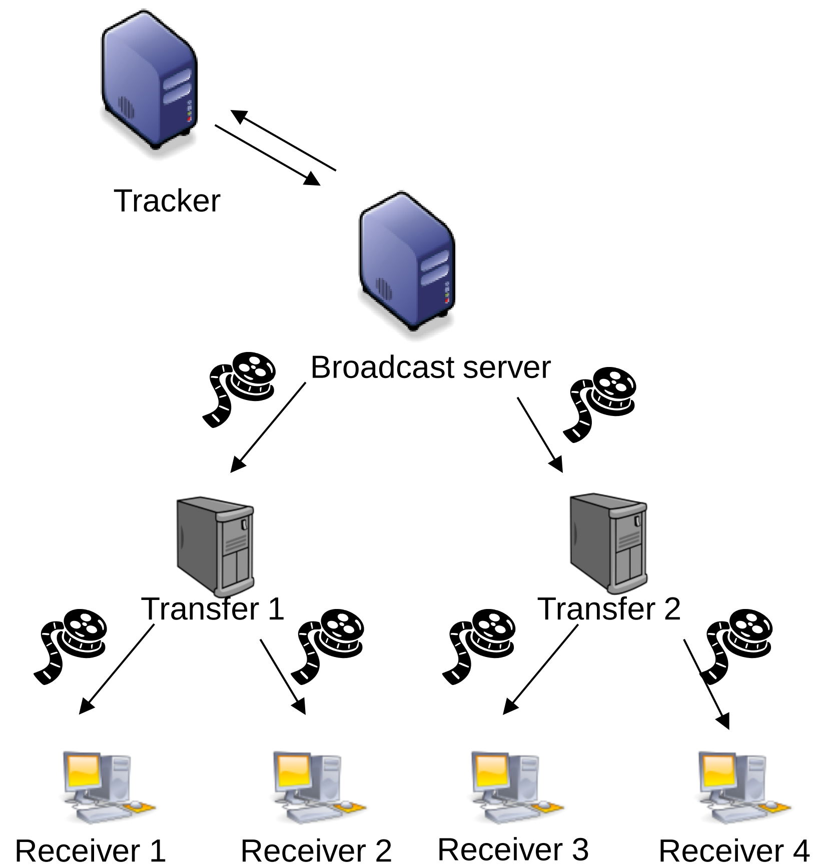
Streaming platform for audio, video, and other streams of data

For each channel a separate tree is constructed

Peers join the Peercast system according to scheme presented before

Content is pushed down to the peers

- No requests are sent by the peers
- Peer only notify parent peer about status



4.3 Adaptive Video Streaming over P2P

TECHNISCHE UNIVERSITÄT DARMSTADT

Motivation

- Streaming of high resolution content costly, requires large CDN
- P2P: works only when there are enough peers online

Solution and vision

- Hybrid solution using adaptive video streaming

Adaptive video streaming

- Multi-layer codec (e.g.: H264 Advanced Video Coding)
  - Base layer always needed (streamed from server)
  - More layers → higher quality
    - Support heterogeneity
    - Always on service and scalable

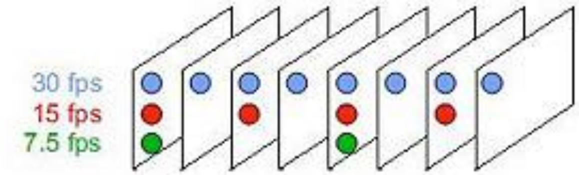
Underlay-aware overlay construction: delay, locality, bandwidth

- How to gather the required information about the underlay
  - NGN equipment

Scalable Video Coding: Flavors of Scalability

Temporal Scalability: Different frame rate, e.g

- 7.5 fps used in mobile phones.
- 15 fps used in Netbooks, iPhone,...
- 30 fps used in PCs



Spatial Scalability: Different resolution.

- QCIF, CIF, TV

Quality scalability

- Different quality levels.

Quality adaptation approaches

- Initial adaptation
  - Support heterogeneous devices
  - Different memory, processing and bandwidth profiles

- Progressive adaptation
  - Support adaptation to changing underlay conditions
  - Bandwidth, delay,...

Picture sources: http://www.hhi.fraunhofer.de/index.php?id=2767&L=1

Main Challenges for (Adaptive) P2P Streaming

Peer selection

- Select peers that can provide the required quality using the available bandwidth
- From which peers to retrieve a chunk

Coordinate delivery from the peers

- Sender-Receiver coordination
- Which peer to give chunks?

Chunk selection

- Which chunks to request / to give?

Handle unpredictable variations in bandwidth

- Using fast layer switching
- What to do if chunks are missing at playback time?