Peer-to-Peer (P2P) Architectures

ECE/CS 658 – Internet Engineering

Dilum Bandara
dilumb@engr.colostate.edu

Outline

- Background
- Unstructured P2P
  - Napster, Gnutella, & BitTorrent
- Structured P2P
  - Chord & Kademlia
- P2P streaming
  - Tree-push approach
  - Mesh-pull approach
    - Chunk scheduling
- Next lab…

P2P - Background

www.associatedcontent.com

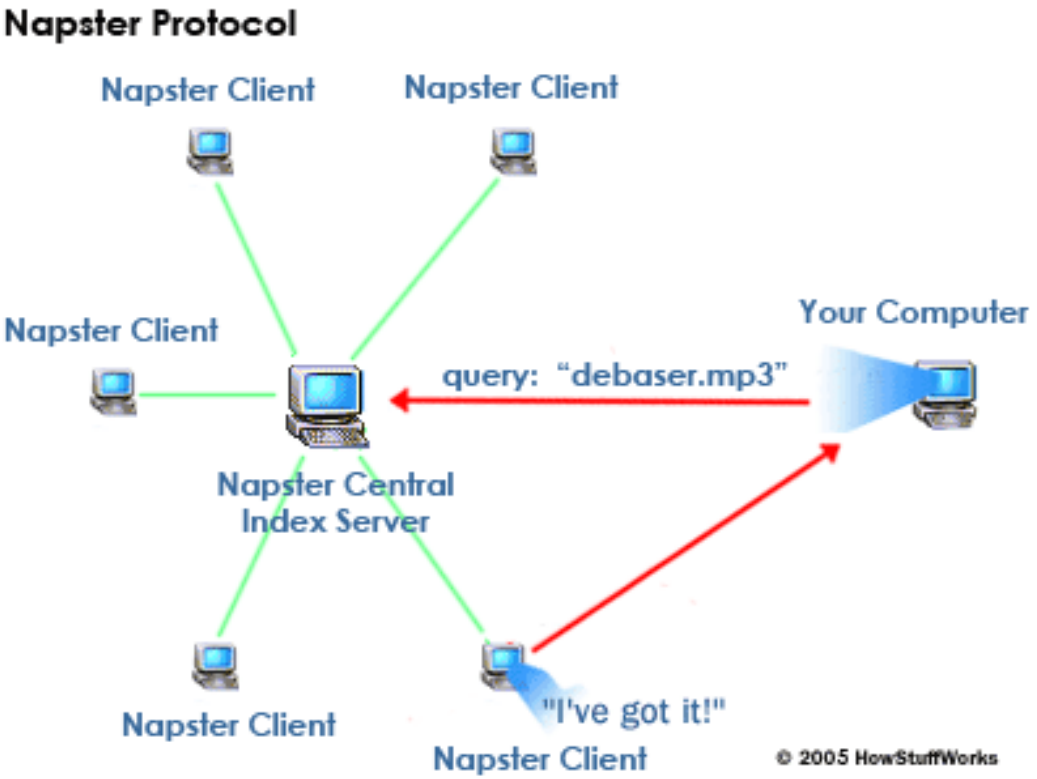
- A distributed system without any central control
- Peers are equivalent in functionality
  - One or more special peers to manage membership
- Tit-for-tat strategy
- Many application domains
  - File sharing – BitTorrent, KaZaA, Napster, BearShare
  - IPTV – PPLive, CoolStreaming, SopCast
  - VoIP – Skype
  - CPU cycle sharing – SETI, World Community Grid
- Middleware - JXTA, MSN P2P
- In 2004, P2P contributed to 50-80% of the Internet traffic
  - Still the volume is same

P2P application characteristics

- Tremendous scalability
  - Millions of peers
  - Anywhere in the world
- Upload/download
- Peers directly talk to each other
- Bandwidth intensive
  - Many concurrent connections
  - Aggressive/unfair bandwidth utilization
- Superpeers
  - Critical for performance/functionality
- Heterogeneous
- Peer churn & failure

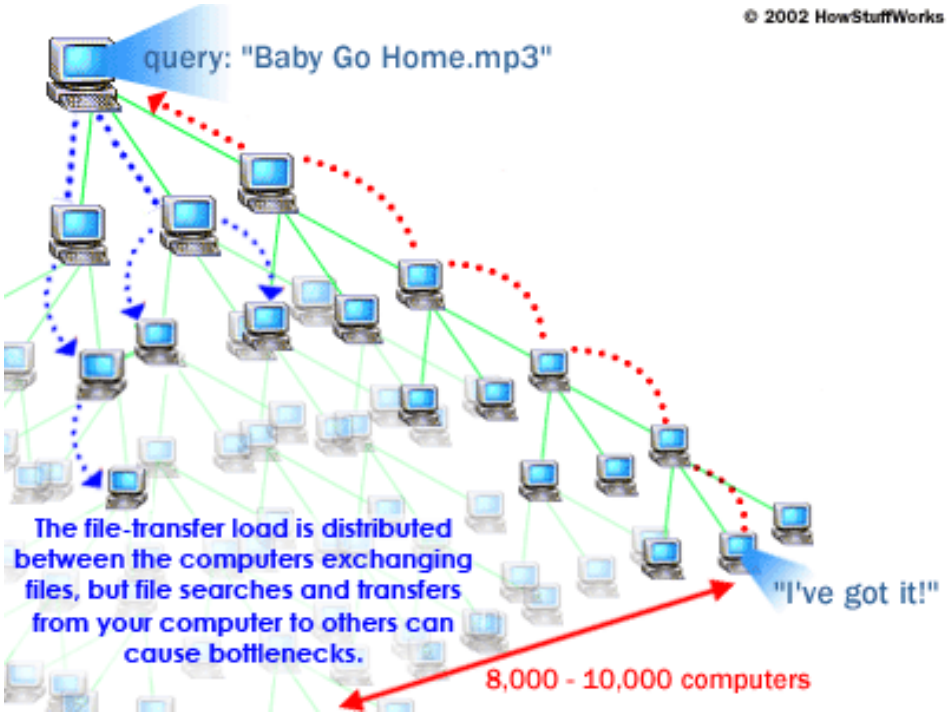
Napster protocol

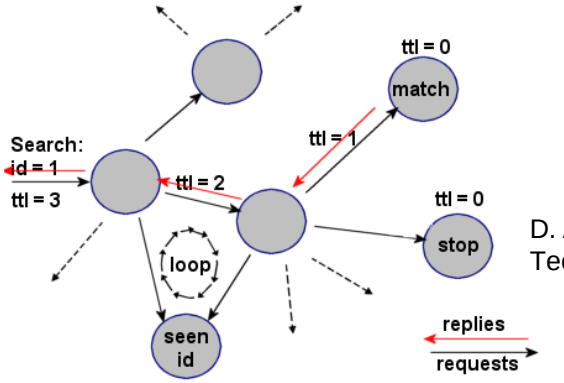
- Killer P2P application
  - June 1999 – July 2001
- 26.4 million users (peek)
- Centralized
  - Guaranteed content discovery
  - Not scalable
  - Easy to track
- Inspired many modern P2P systems



Gnutella protocol

- Fully distributed
- Initial entry point is known
- Maintain dynamic list of partners
- Flooding
  - Guaranteed content discovery
  - Not scalable
  - Harder to track
- TTL based random walk
  - Content discovery is not guaranteed
- Today is more of a protocol than a client

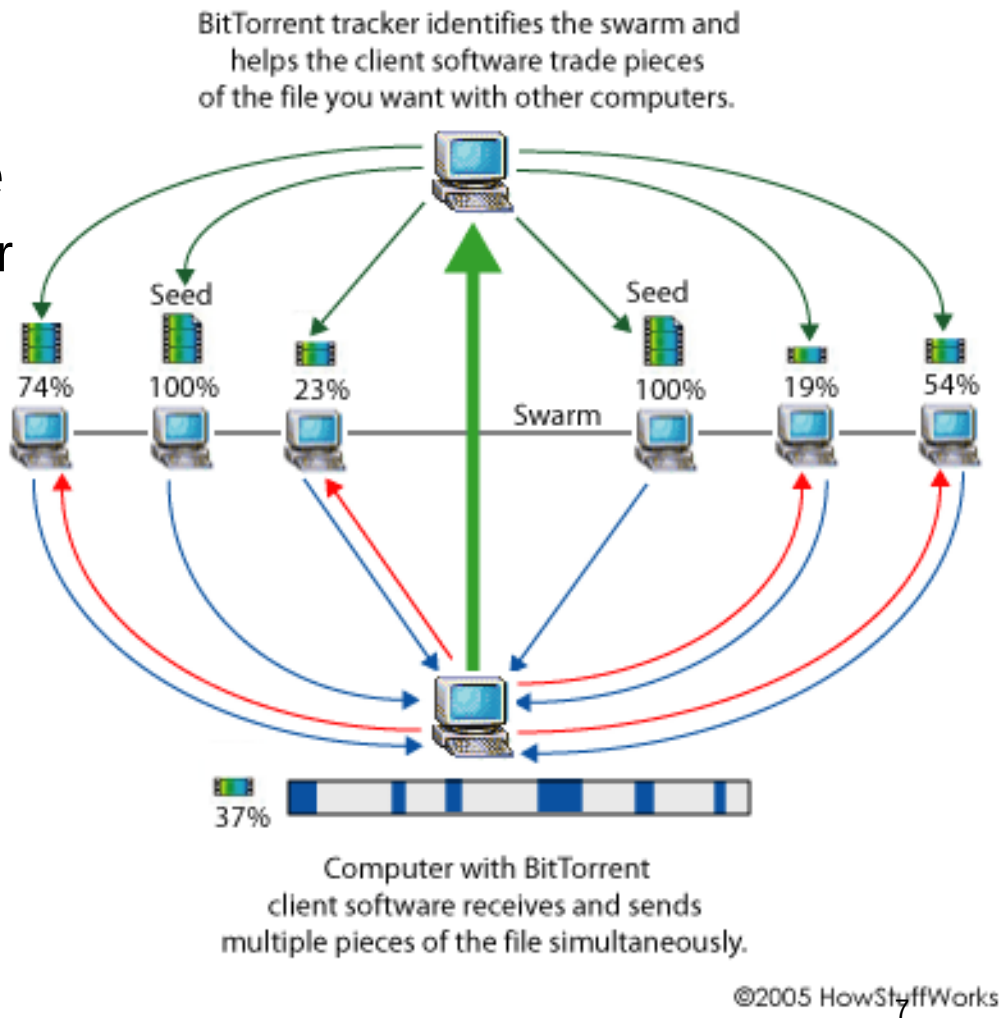




D. Aitken et al., "Peer-to-Peer Technologies and Protocols"

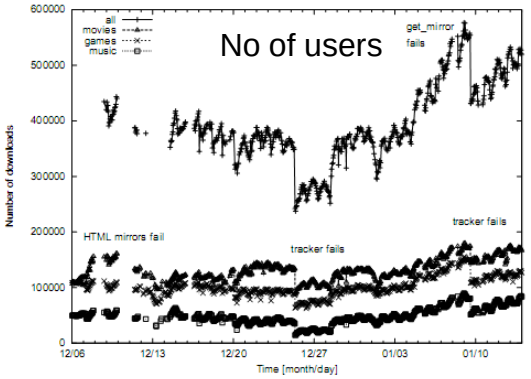
BitTorrent protocol

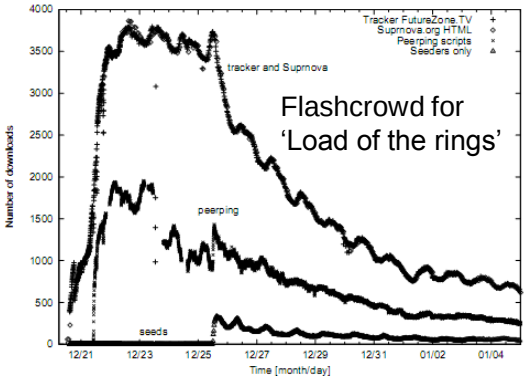
- Content owner publish an URL to a web site
- URL points to a *.torrent* file
  - Stored in a .torrent file server
- *.torrent* file points to a tracker(s)
  - Registry of leaches & seeds for a given file
- Tracker give a random list of peer IP addresses
- Files are shared based on chunk IDs
- Enforce fairness

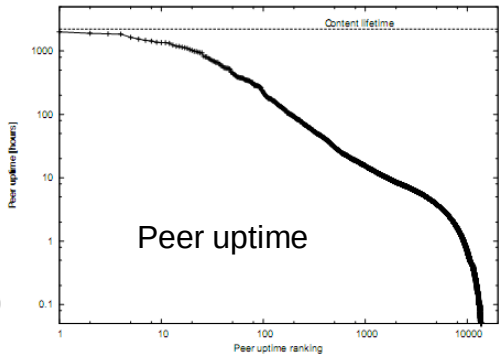


BitTorrent protocol (cont.)

- Many websites with URL databases – communities
  - Guaranteed to find content if published
- Trackers are replicated
  - Harder to track
  - You can run your own tracker
- Upload already downloaded chunks
- Tit-for-tat
  - Give to 4 peers that give me the highest download bandwidth
  - 1 random peer







(Pouwelse, 2005)

Summary - Unstructured P2P

- Content discovery & delivery are separated
- Content discovery is mostly outside the P2P overlay
- Centralized solutions
  - Not scalable
  - Affect content delivery when failed
- Distributed solutions
  - High overhead
  - May not locate the content
- No predictable performance
  - Delay or message bounds

Outline

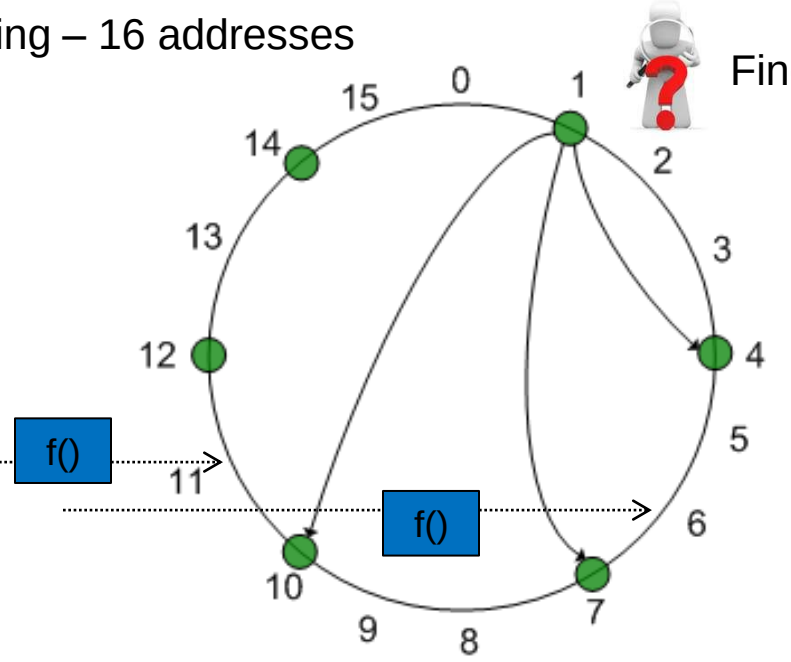
- Background
- Unstructured P2P
  - Napster, Gnutella, & BitTorrent
- Structured P2P
  - Chord & Kademlia
  - Broadcasting with Chord
- P2P streaming
  - Tree-push approach
  - Mesh-pull approach
    - Chunk scheduling
- Next lab…

Structured P2P

- A deterministic way to locate contents & peers
- Locate the peer responsible for a given key
- Key – hash
  - 128-bit or higher
  - Hash of file name, metadata, or actual content
- Peers also have a key
  - Random bit string or IP address
- Distributed Hash Tables (DHTs) are used index keys
- Node responsible for the key stores a pointer to the peer having content
  - Pointer - IP address & port number
- Same protocol is used to publish & locate content

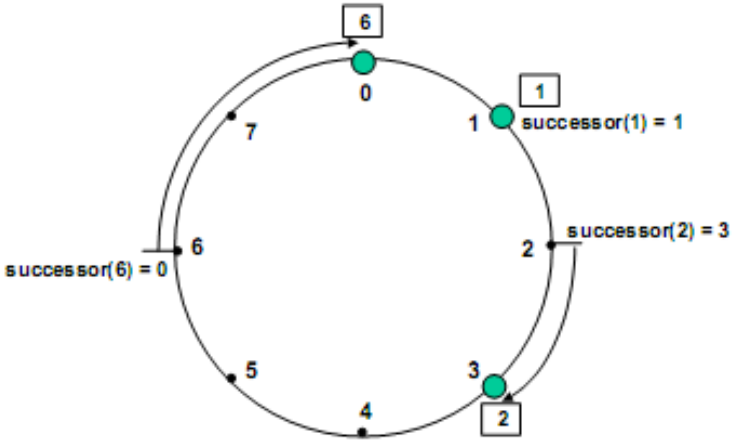
Example

Song.mp3
Cars.mpeg

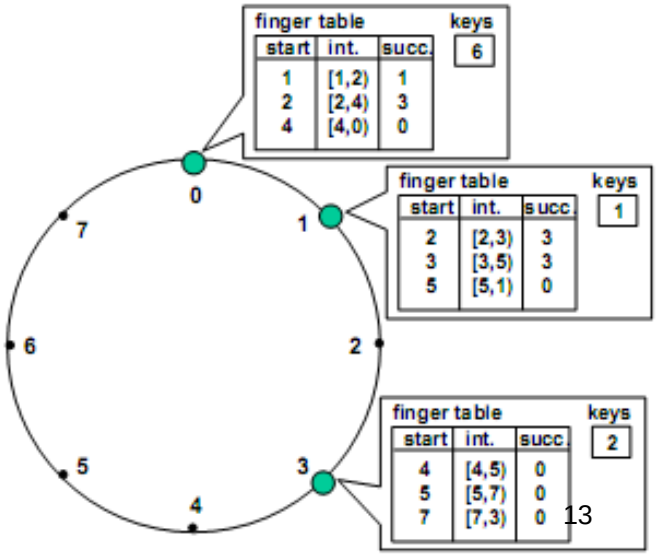


Chord

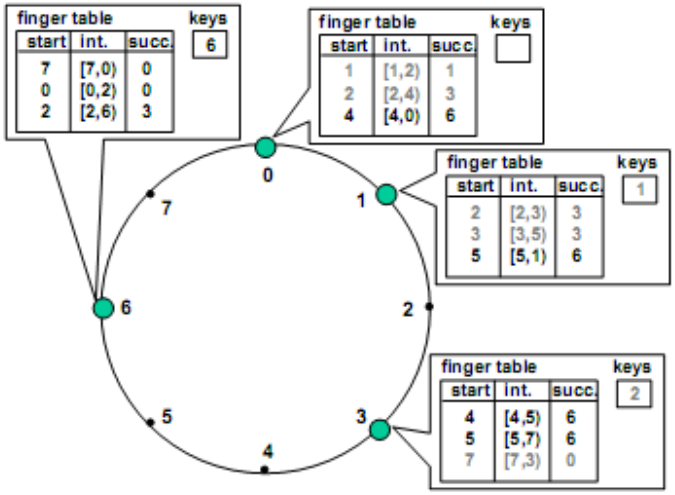
- Key space is arranged as a ring
- Each peer is responsible for portion of the ring
  - Called the successor of a key
  - 1st peer in clockwise direction
- Routing table
  - Keep a pointer (finger) to m peers
  - Keep a finger to 2^{i-1}-th peer, $1 \leq i \leq m$
- Key resolution
  - Go to the peer with the closest key
  - Recursively continue until key is find
  - Can be located within $O(\log n)$ messages



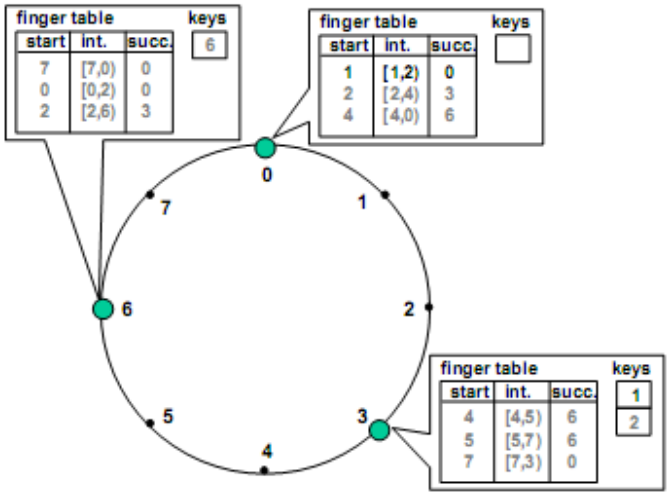
m =3-bit key ring



Chord (cont.)



New peer with key 6 joins the overlay



Peer with key 1 leave the overlay

- New peer entering the overlay
  - Takes keys from the successor
- Peer leaving the overlay
  - Give keys to the successor
- Peer failure or churn makes finger table entries stale

Kademlia

- Used in eMule, aMule, & AZUREUS
- 160-bit keys
  - Nodes are assigned random keys
- Distance (closeness) between 2 keys is determined by XOR
  - Routing in the ring is bidirectional
- Keys are stored in nodes with the shortest XOR distance
- k-bucket routing table
  - Store up to k peers for each distance between $(2^i, 2^{i+1})$
  - Learn about new peers from queries
  - Update bucket entries based on least-recently seen approach
  - Ping a node before dropping from a bucket
  - Better performance under peer churn & failure

Kademlia (cont.)

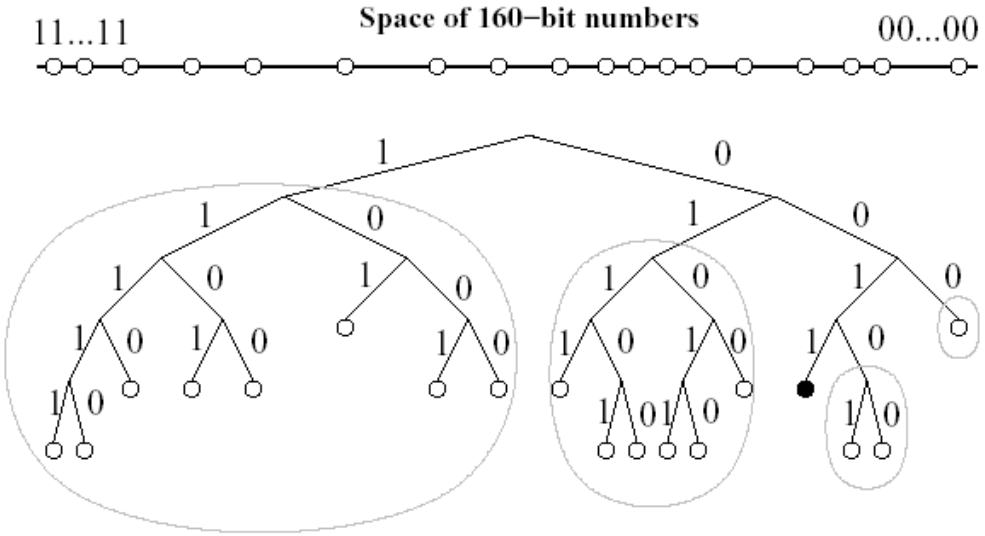


Fig. 1: Kademlia binary tree. The black dot shows the loation of node 0011... in the tree. Grey ovals show subtrees in which node 0011... must have a contact.

Node with key 0011 keeps k entries for

- 1000/1
- 0100/2
- 0000/3
- 0010/4

(Bland et al., P2P Routing)

- Routing
  - Find a set of peers with the lowest distance in routing table
    - Match the longest prefix
  - Concurrently ask α of them to find an even closer peer
  - Iterate until no closer peers can be found
  - Then send the query to α closest peers

Summary - Structured P2P

- Content discovery is within the P2P overlay
- Deterministic performance
- Chord
  - Unidirectional routing
  - Recursive
  - Peer churn & failure is an issue
- Kademlia
  - Bidirectional routing
  - Iterative
  - Can work even with peer failure & churn
- MySong.mp3 is not same as mysong.mp3
- Unbalanced distribution of keys

Comparison

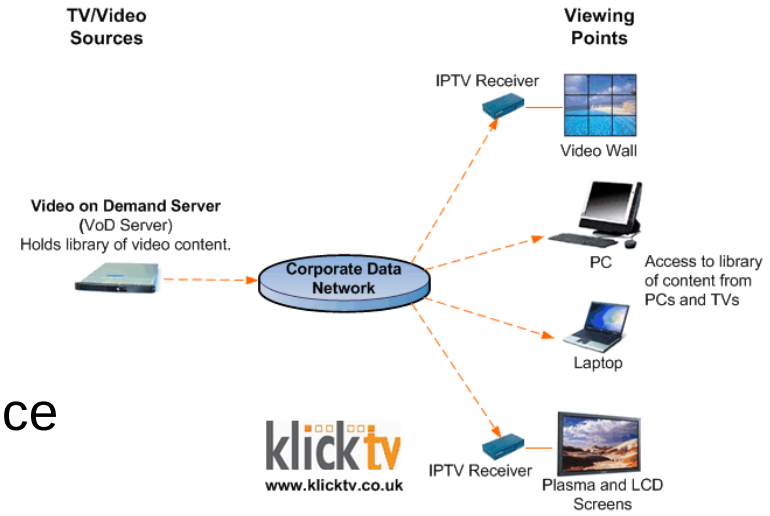
	Unstructured P2P	Structured P2P
Overlay construction	High flexibility	Low flexibility
Resources	Indexed locally	Indexed remotely on a distributed hash table
Query messages	Broadcast or random walk	Unicast
Content location	Best effort	Guaranteed
Performance	Unpredictable	Predictable bounds
Overhead	High	Relatively low
Object types	Mutable, with many complex attributes	Immutable, with few simple attributes
Peer churn & failure	Supports high failure rates	Supports moderate failure rates
Applicable environments	Small-scale or highly dynamic, e.g., mobile P2P	Large-scale & relatively stable, e.g., desktop file sharing
Examples	Gnutella, LimeWire, KaZaA, BitTorrent	Chord, CAN, Pastry, eMule, BitTorrent

Outline

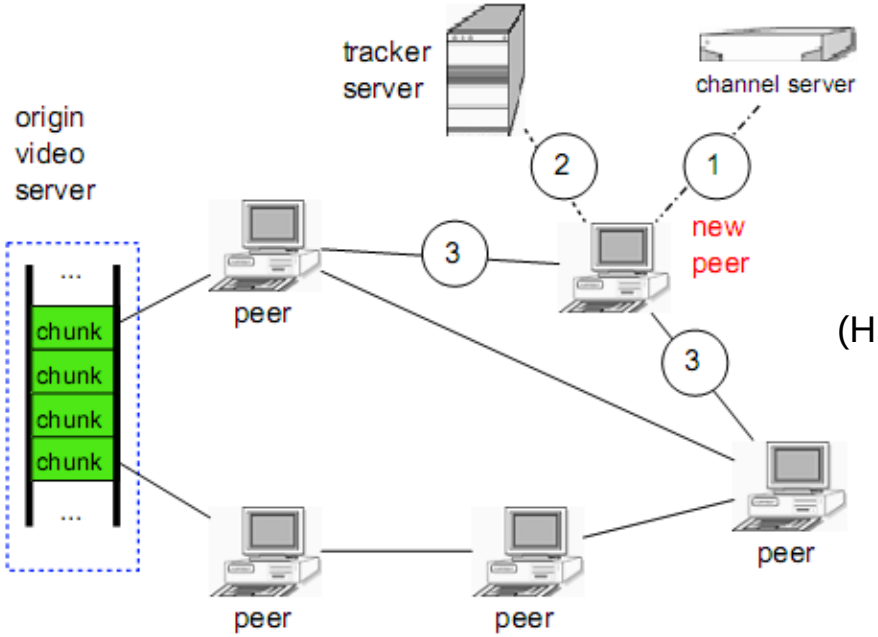
- Background
- Unstructured P2P
  - Napster, Gnutella, & BitTorrent
- Structured P2P
  - Chord & Kademlia
  - Broadcasting with Chord
- P2P streaming
  - Tree-push approach
  - Mesh-pull approach
    - Chunk scheduling
- Next lab…

P2P streaming

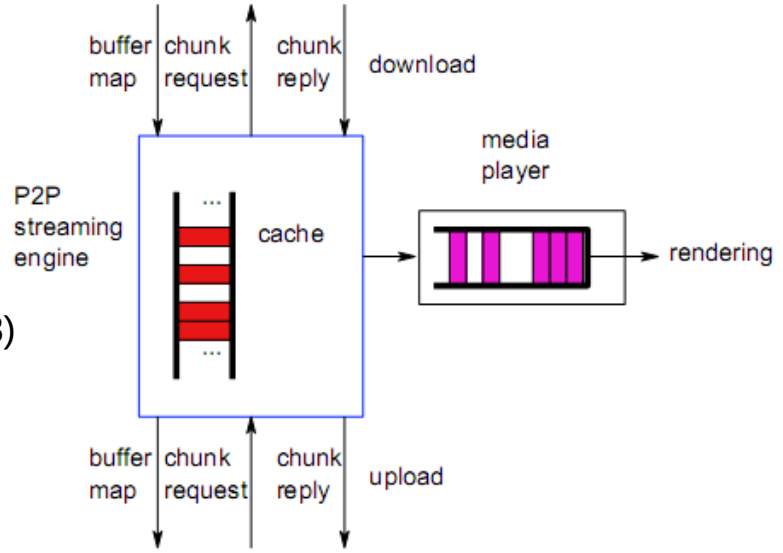
- Emergence of IPTV
  - Content Delivery Networks (CDNs) can't handle the bandwidth requirements
  - No multicast support at network layer
- P2P
  - Easy to implement
    - No global topology maintenance
  - Tremendous scalability
    - Greater the demand better the service
  - Robustness
    - No single point of failure
    - Adaptive
  - Application layer
  - Cost effective



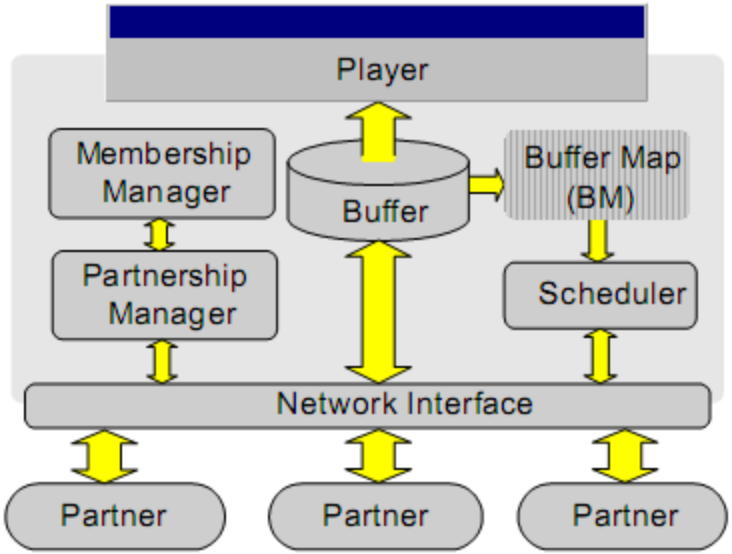
P2P streaming components



(Hie, 2008)



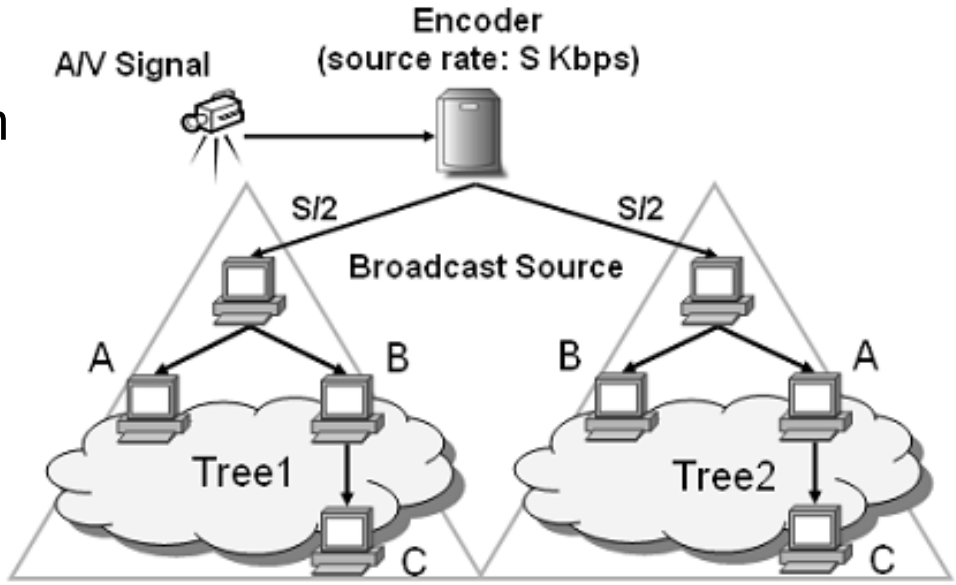
- Chunk – segment of the stream
  - E.g., one second worth of video



(Zhang, 2005)

Tree-push approach

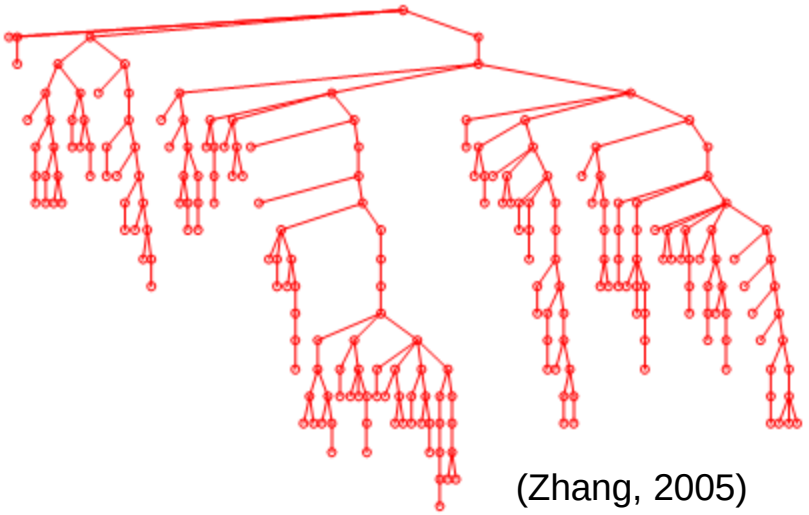
- Overlay tree is constructed starting from the source
- Parent selection can be based on
  - Bandwidth, latency, number of peers, etc.
- Data is pushed down the tree from a parent to child peers
- Multi-tree based approach
  - For better content distribution
  - For reliability



(Liu, 2008)

Tree-push approach - Issues

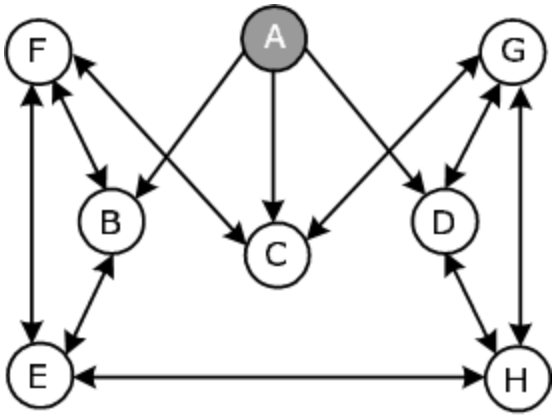
- Connectivity is affected when peers at the top of the hierarchy leave or fail
- Time to reconstruct the tree
- Unbalanced tree
- Majority of the peers are leaves
  - Unable to utilize their bandwidth
- Mesh is more robust than a tree



(Zhang, 2005)

Mesh-pull approach

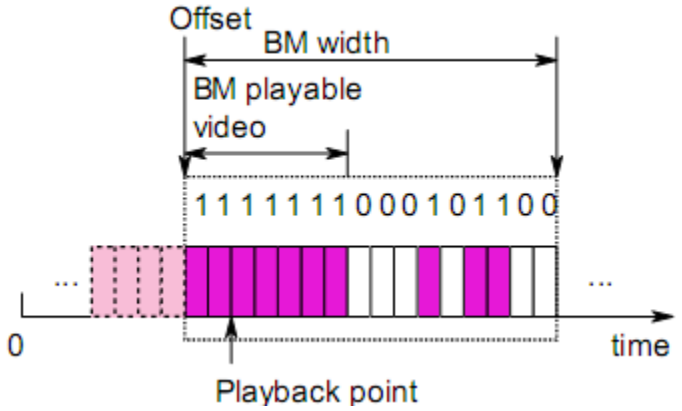
- A peer connects to multiple peers
- Pros
  - More robust to failure
  - Better bandwidth utilization
- Cons
  - No specific chunk forwarding path
  - Need to pull chunks from partners
  - Need to know which partner has what
- Most commercial products use this approach



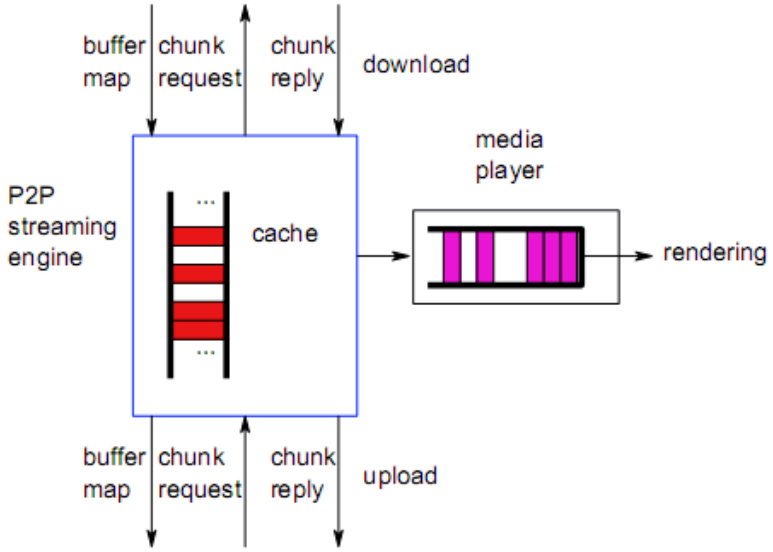
(Zhang, 2005)

Chunk sharing

- Each peer
  - Caches a set of chunks within a sliding window
  - Shares its chunk information with its partners
- Buffer maps are used to inform chunk availability
- Chunks may be in one or more partners
- What chunks to get from whom?



(Hie, 2008)



Chunk scheduling

- Some chunks are highly available while others are scare
- Some chinks needs to be played soon
- New chunks need to be pulled from video source
- Chunk scheduling consider how a peer can get chunks while
  - minimizing latency
  - preventing skipping
  - maximizing throughput
- Determines the user Quality of Experience (QoE)
- Most commercial products use TCP for chunk transmission
- Chunk scheduling
  - Random, rarest first, earliest deadline first, earliest deadline & rarest first

P2P issues & opportunities

Issues

- Peer churn & failure
- Heterogeneous upload bandwidth
- Who is willing to be a superpeer
- Startup delay
- Not really real-time
- Supporting different video qualities
- Flashcrowds
- NATs & firewalls
- Digital rights management

Opportunities

- Resource discovery
- Better peering strategies
- Better bandwidth utilization/adaptation
- Robust topology construction
- Supporting variable bit rates
- Network level content caching
- Traffic identification & control
- Is TCP/UDP is the best?

Summary

- P2P systems are highly scalable
- Content discovery is still not optimum
- Peer churn & failure is a problem
  - Both for structured & unstructured
- P2P streaming
  - Mesh-pull approach is more robust
  - Scheduling algorithm determines the user QoE

Next Lab

- P2P-based content discovery system
  - Static content discovery using Chord
    - E.g., file names, CPU speed, etc.
  - Register file names
  - Query for file names
  - Read Chord paper
- Each student is responsible for 4 peers on PlanetLab
- Inter operability is the key
  - Content registering/searching protocol format will be given
- Peer host by the TA will be the entry point to the system
- Following lab will be built on this

Bibliography

1. D. Aitken, J. Bligh, O. Callanan, D. Corcoran, and J. Tobin “Peer-to-peer technologies and protocols,” Available: http://ntrg.cs.tcd.ie/undergrad/4ba2.02/p2p/index.html
2. R. Bland, D. Caulfield, E. Clarke, A. Hanley, and E. Kelleher, “P2P routing,” Available: http://ntrg.cs.tcd.ie/undergrad/4ba2.02-03/p9.html
3. Z. Chen, K. Xue, and P. Hong, “A study on reducing chunk scheduling delay for mesh-based P2P live streaming,” In Proc. of 7th International Conference on Grid and Cooperative Computing, 2008, pp. 356-361.
4. X. Hei, Y. Liu, and K. W. Ross, “IPTV over P2P streaming networks: the mesh-pull approach,” IEEE Communications Magazine, vol. 46, no. 2, Feb. 2008, pp. 86-92.
5. V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy and A. E. Mohr, “Chainsaw: eliminating trees from overlay multicast,” In Proc. of 4th International Workshop on Peer-to-Peer Systems (IPTPS), Feb. 2005, pp. 127-140.
6. J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips, “The bittorrent p2p file-sharing system: measurements and analysis,” In Proc 4th International Workshop on Peer-to-Peer Systems (IPTPS), 2005.
7. J. Liu, S. G. Rao, B. Li, and H. Zhang, “Opportunities and challenges of peer-to-peer internet video broadcast,” In Proc. of IEEE, vol. 96, no. 1, Jan. 2008, pp. 11-24.
8. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: a scalable peer-to-peer lookup service for internet applications,” In Proc. ACM SIGCOMM '01, San Diego, CA, pp. 149-160.
9. D. Talia, P. Trunfio, J. Zeng, and M. Hogqvist, “A DHT-based peer-to-peer framework for resource discovery in grids," CoreGRID Technical Report, No TR-0048, June 2006.
10. X. Zhang, J. Liu, B. Li, and T. P. Yum, “CoolStreaming/DONet: a data-driven overlay network for efficient live media streaming,” In Proc. of INFOCOM 2005, Miami, USA, Mar. 2005.
11. M. Zhang, Y. Xiong, Q. Zhang, and S. Yang, “On the optimal scheduling for media streaming in data-driven overlay networks,” In Proc. of Global Telecommunications Conference (GLOBECOM 06), San Francisco, USA, Nov.-Dec. 2006.

Questions/Comments

Thank you!