ECEN 757
P2P Systems

Chapter 10

Midterm exam for Remote Session

- Midterm will be conducted remotely over Canvas/Examity
- You can take the exam between 3/28 6pm to 4/1 midnight
- Exam time = 90 minutes (75 minutes for answering the problems and 15 minutes for scanning and uploading)
- Do NOT attend the class on 3/28 even if you are on campus
- Please practice Exam 0 in advance

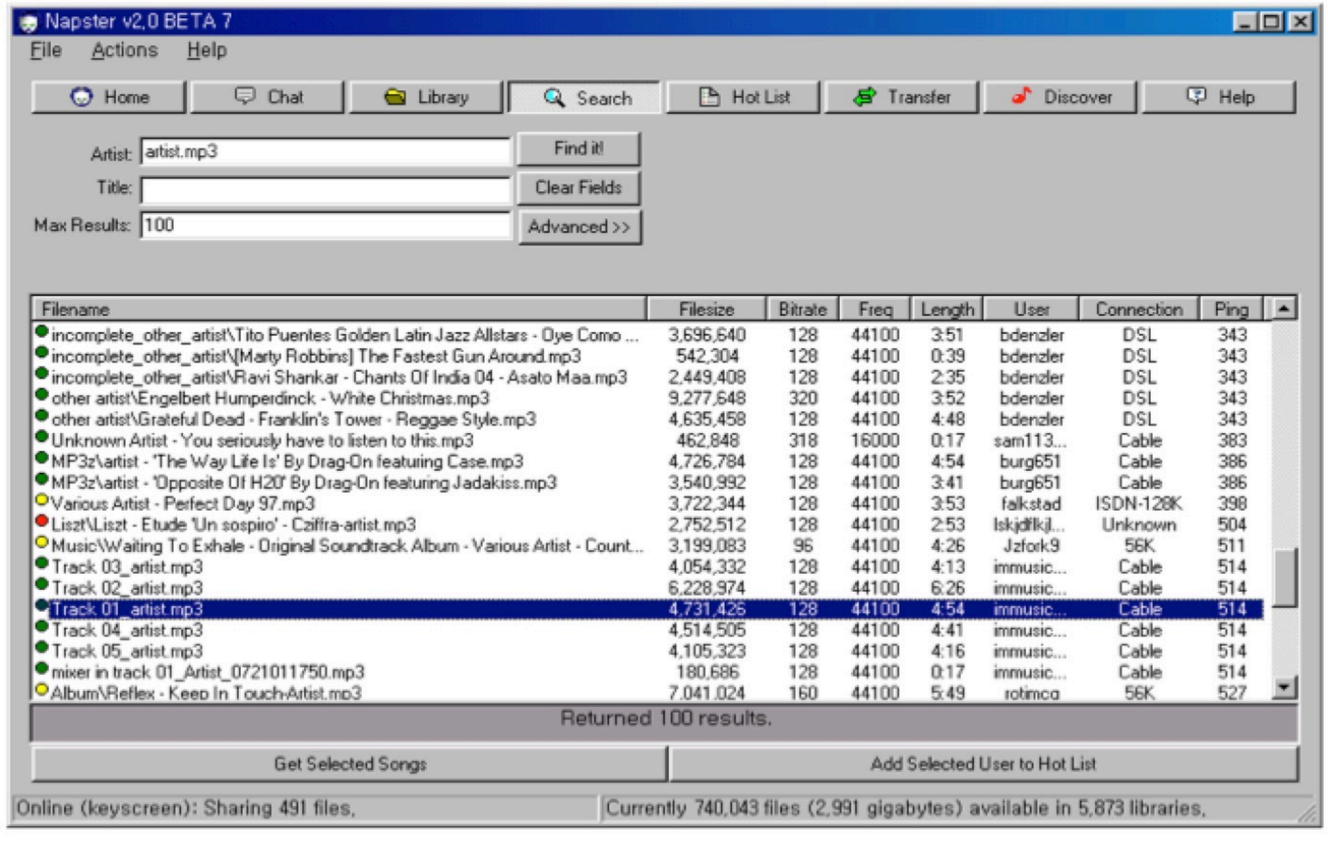
Announcement about communications

- Please email me (ihou@tamu.edu) for any issues
- Do NOT leave comments on Canvas… I do not read them

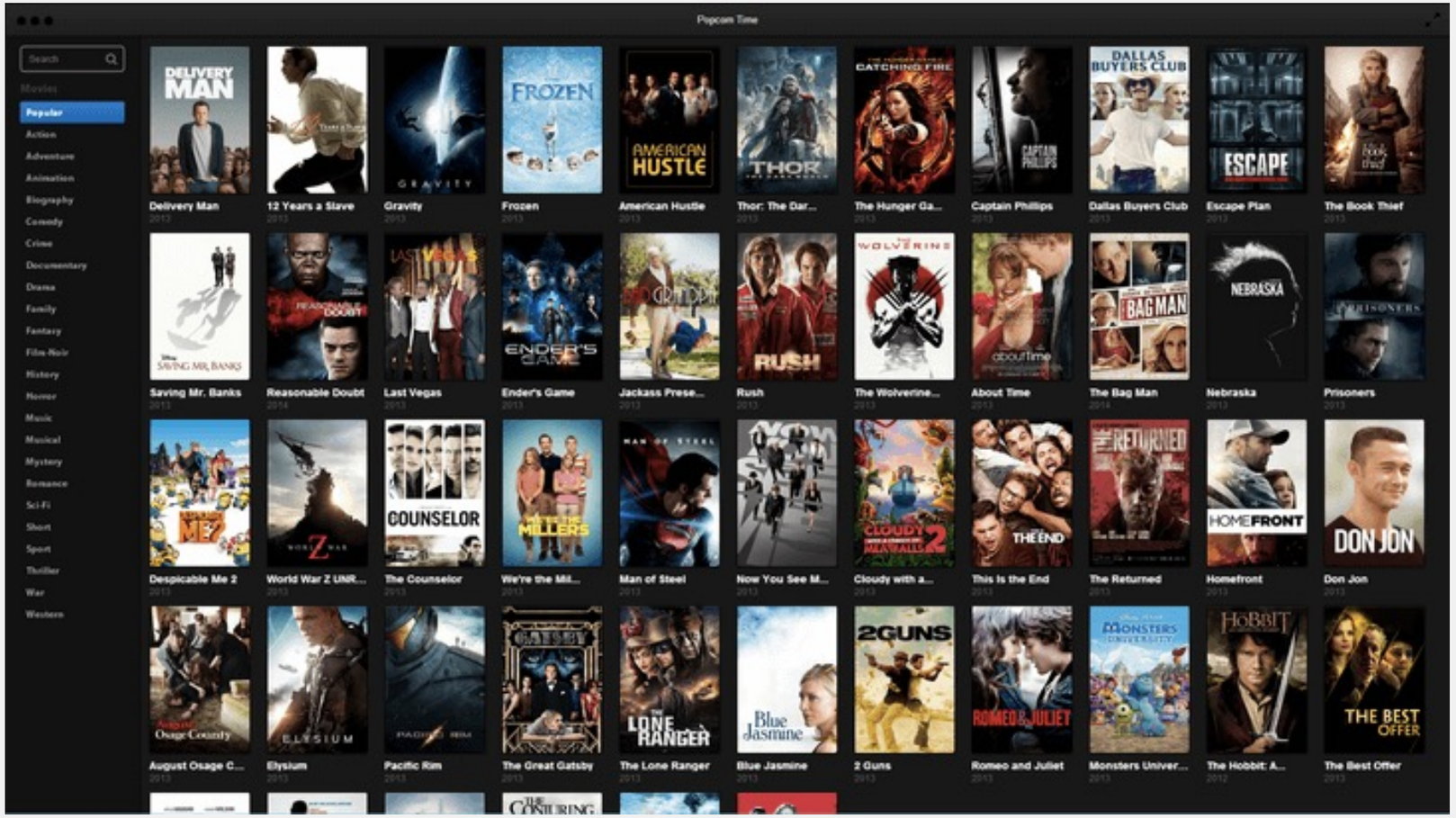
Why Study Peer to Peer Systems?

- First distributed systems that seriously focused on scalability with respect to number of nodes
- P2P techniques abound in cloud computing systems
    - Key-value stores (e.g., Cassandra, Riak, Voldemort) use Chord p2p hashing

Why Study Peer to Peer Systems?



Why Study Peer to Peer Systems?



A Brief History

- [6/99] Shawn Fanning (freshman Northeastern U.) releases Napster online music service
- [12/99] RIAA sues Napster, asking $100K per download
- [3/00] 25% UWisc traffic Napster, many universities ban it
- [00] 60M users
- [2/01] US Federal Appeals Court: users violating copyright laws, Napster is abetting this
- [9/01] Napster decides to run paid service, pay % to songwriters and music companies
- [Today] Napster protocol is open, people free to develop opennap clients and servers http://opennap.sourceforge.net
  - Gutella: http://www.limewire.com (deprecated)
  - Peer to peer working groups: http://p2p.internet2.edu

What We Will Study

- Widely-deployed P2P Systems
  1. Napster
  2. Gnutella
  3. Fasttrack (Kazaa, Kazaalite, Grokster)
  4. BitTorrent
- P2P Systems with Provable Properties
  1. Chord
  2. Pastry
  3. Kelips

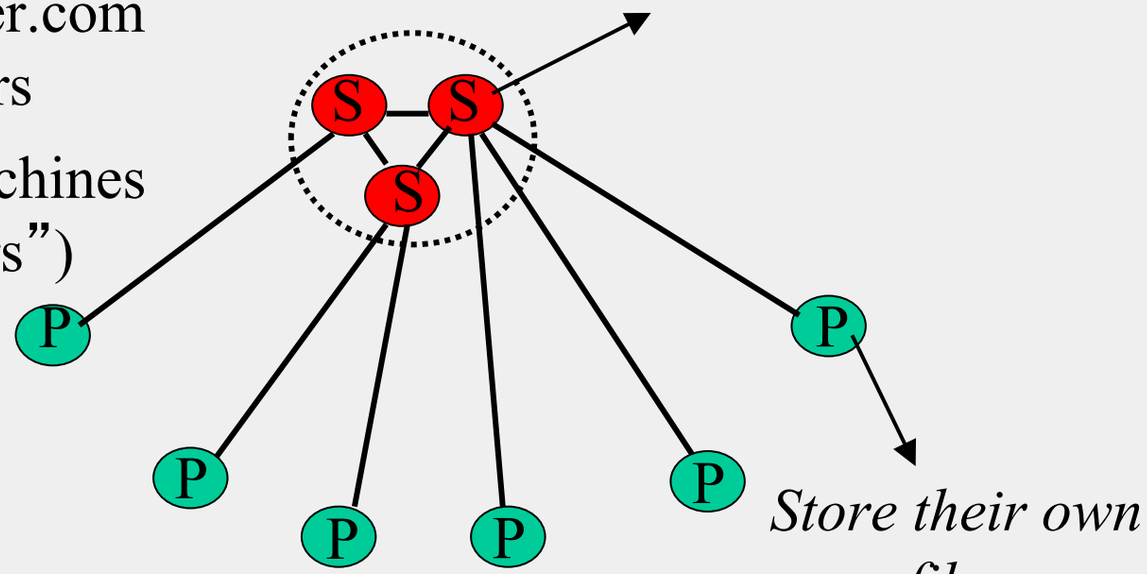
Napster Structure

Filename	Info about
PennyLane.mp3	Beatles, @ 128.84.92.23:1006

Store a directory, i.e., filenames with peer pointers

napster.com Servers

Client machines ("Peers")



Store their own files

Napster Operations

Client

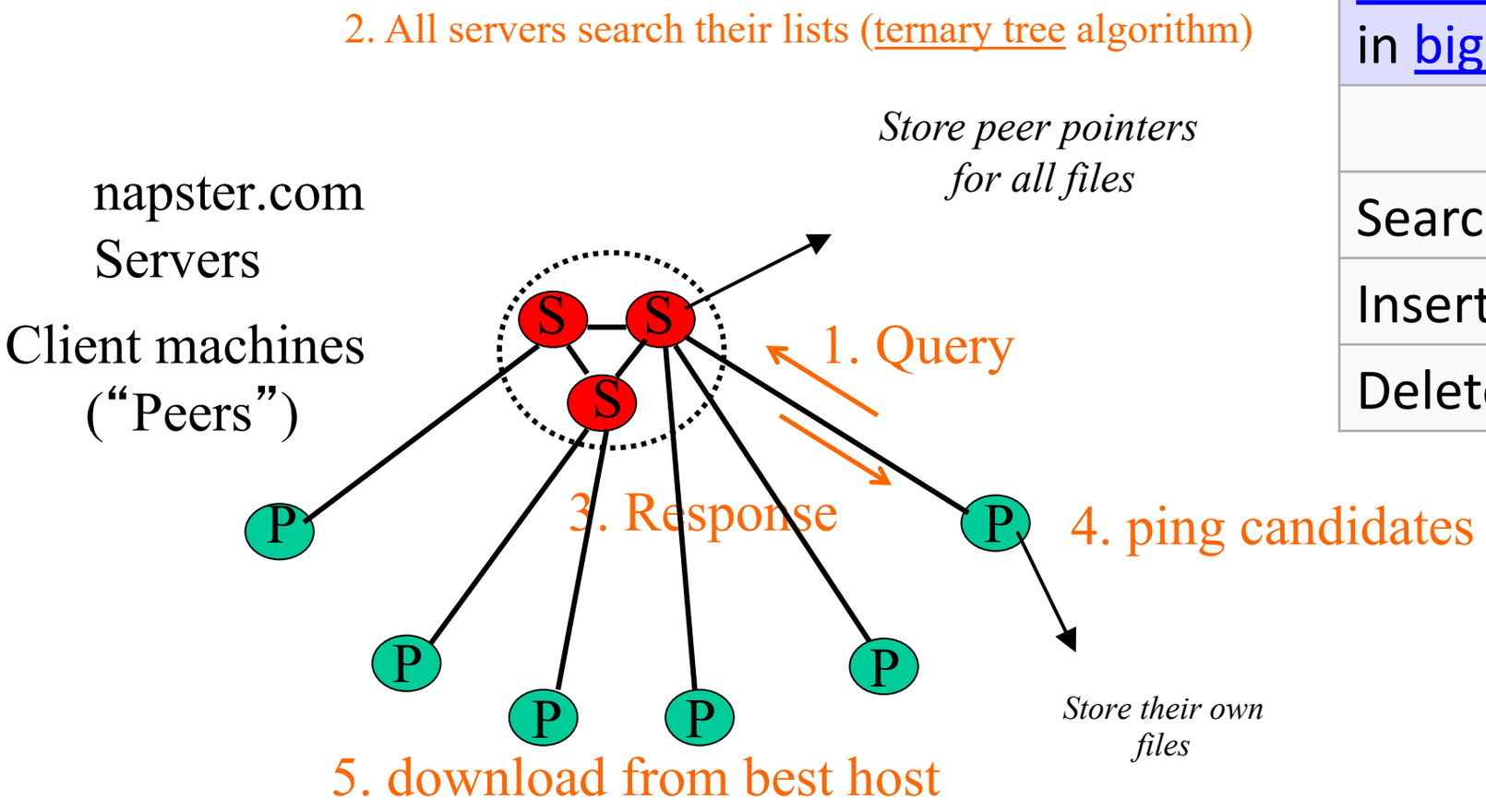
- Connect to a Napster server
  - Upload list of music files that you want to share
  - Server maintains list of <filename, ip_address, portnum> tuples. **Server stores no files.**

Napster Operations

Client (contd.)

- Search
  - Send server keywords to search with
  - (Server searches its list with the keywords)
  - Server returns a list of hosts - <ip_address, portnum> tuples - to client
  - Client pings each host in the list to find transfer rates
  - Client fetches file from best host
- All communication uses TCP (Transmission Control Protocol)
  - Reliable and ordered networking protocol

Napster Search



Ternary Search Tree (TST)		
Type	Tree	
Time complexity in big O notation		
	Average	Worst case
Search	O(log *n*)	O(*n*)
Insert	O(log *n*)	O(*n*)
Delete	O(log *n*)	O(*n*)

Joining a P2P system

- Can be used for any p2p system
  - Send an http request to well-known url for that P2P service - `http://www.myp2pservice.com`
  - Message routed (after lookup in DNS=Domain Name system) to introducer, a well known server that keeps track of some recently joined nodes in p2p system
  - Introducer initializes new peers' neighbor table

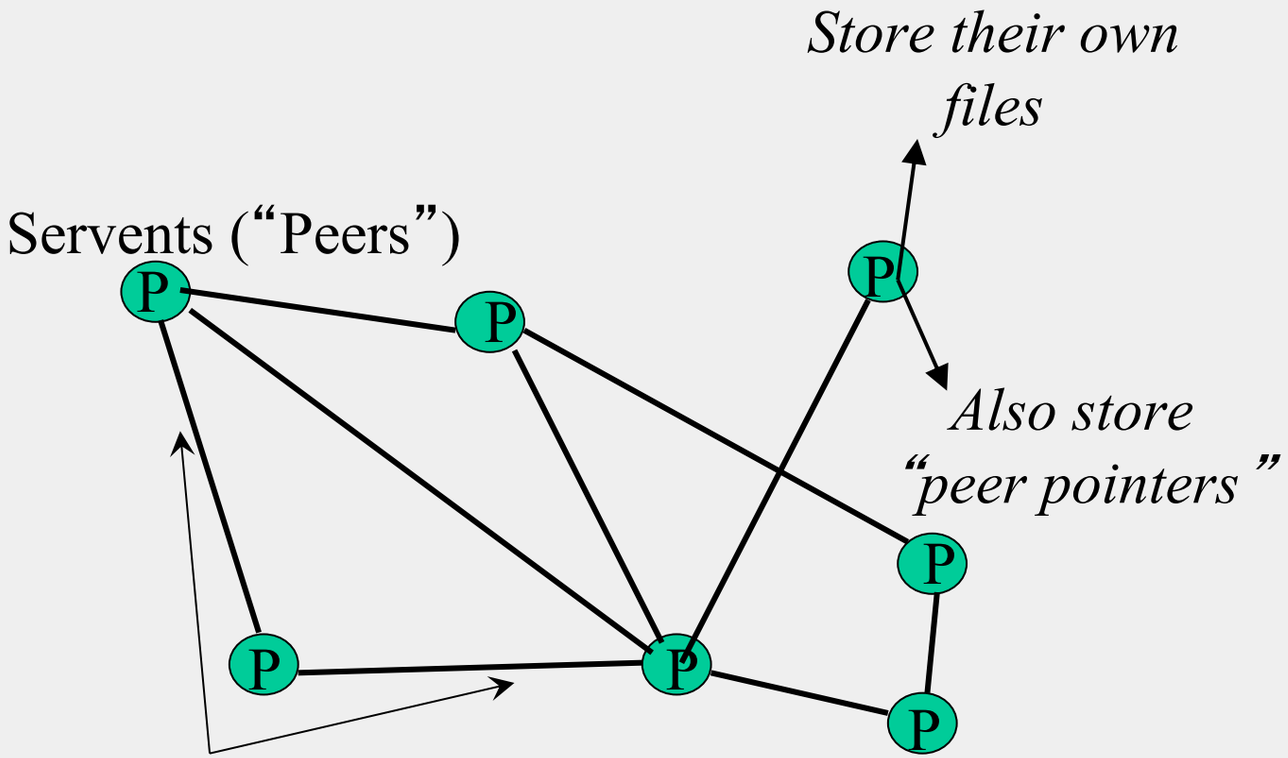
Problems

- Centralized server a source of congestion
- Centralized server single point of failure
- No security: plaintext messages and passwords
- napster.com declared to be responsible for users' copyright violation
  - "Indirect infringement"
  - Next system: Gnutella

Gnutella

- Eliminate the servers
- Client machines search and retrieve amongst themselves
- Clients act as servers too, called **servents**
- [3/00] release by AOL, immediately withdrawn, but 88K users by 3/03
- Original design underwent several modifications

Gnutella



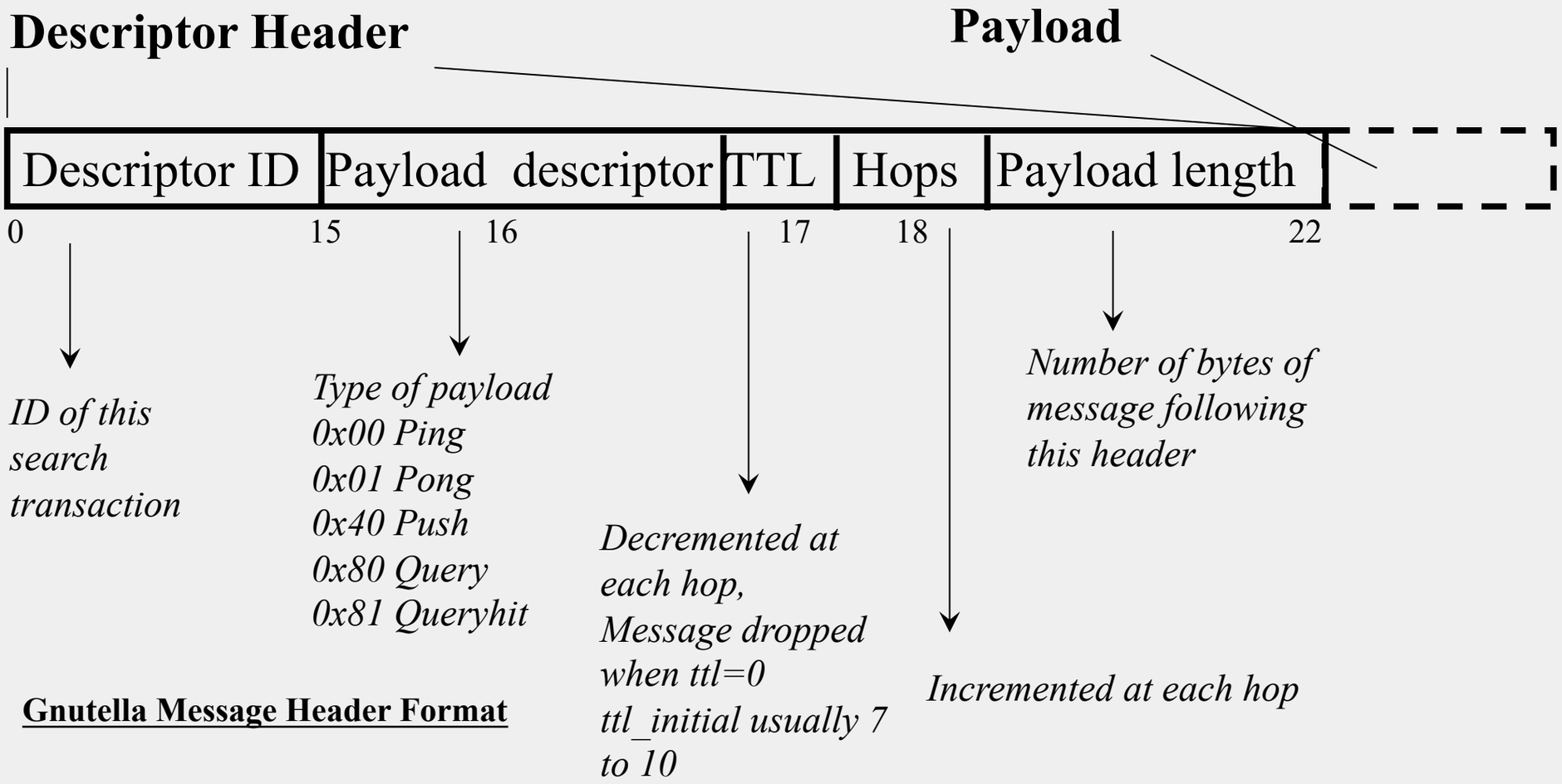
Connected in an **overlay** graph

(== each link is an implicit Internet path)

How do I search for my Beatles file?

- Gnutella *routes* different messages within the overlay graph
- Gnutella protocol has 5 main message types
  - Query (search)
  - QueryHit (response to query)
  - Ping (to probe network for other peers)
  - Pong (reply to ping, contains address of another peer)
  - Push (used to initiate file transfer)
- We'll go into the message structure and protocol now
  - All fields except IP address are in little-endian format
  - Ox12345678 stored as 0x78 in lowest address byte, then 0x56 in next higher address, and so on.

How do I search for my Beatles file?



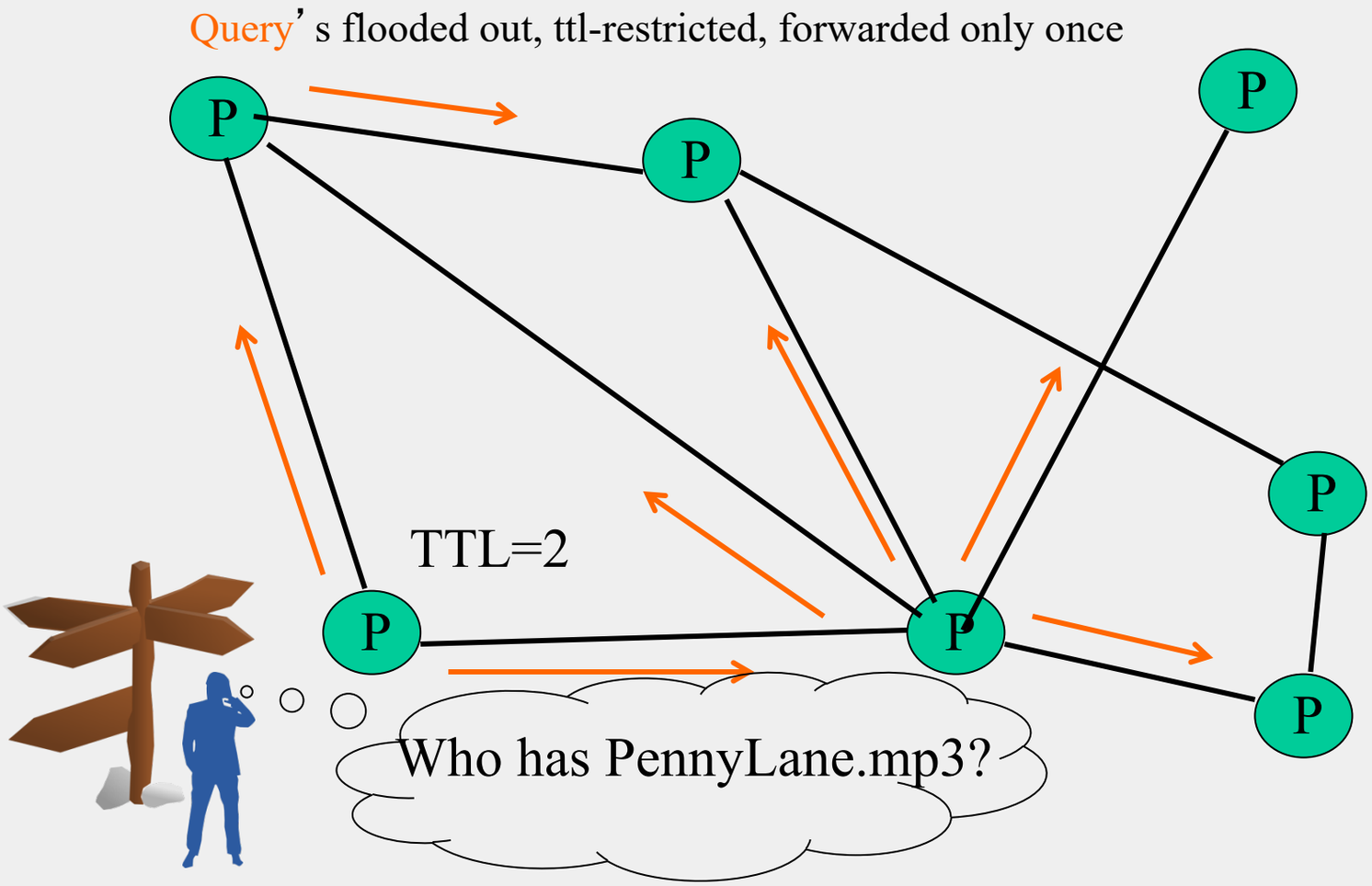
Gnutella Message Header Format

How do I search for my Beatles file?

Query (0x80)

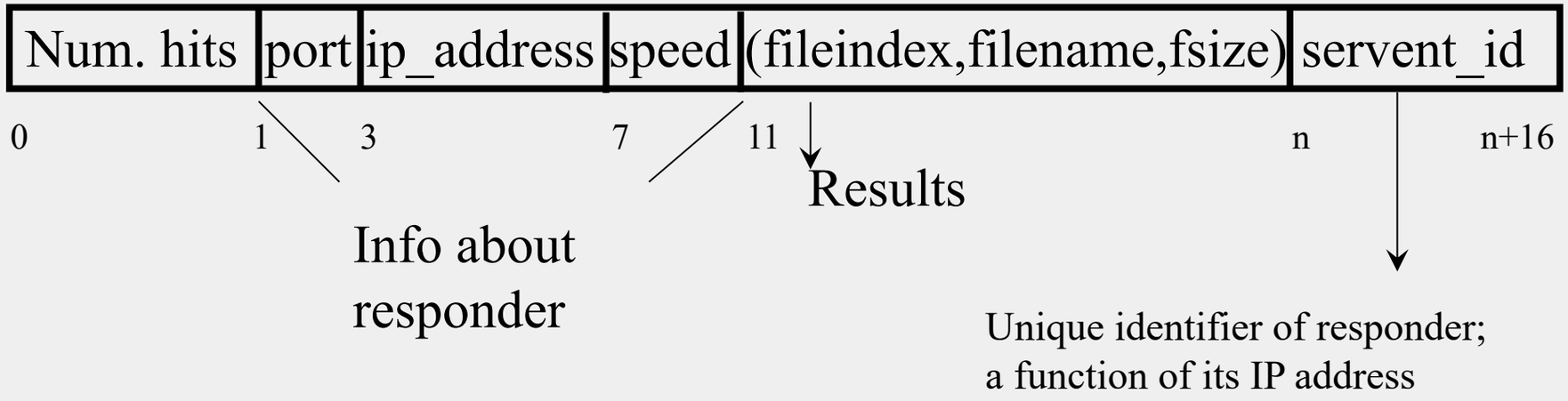
Payload Format in Gnutella Query Message

Gnutella Search



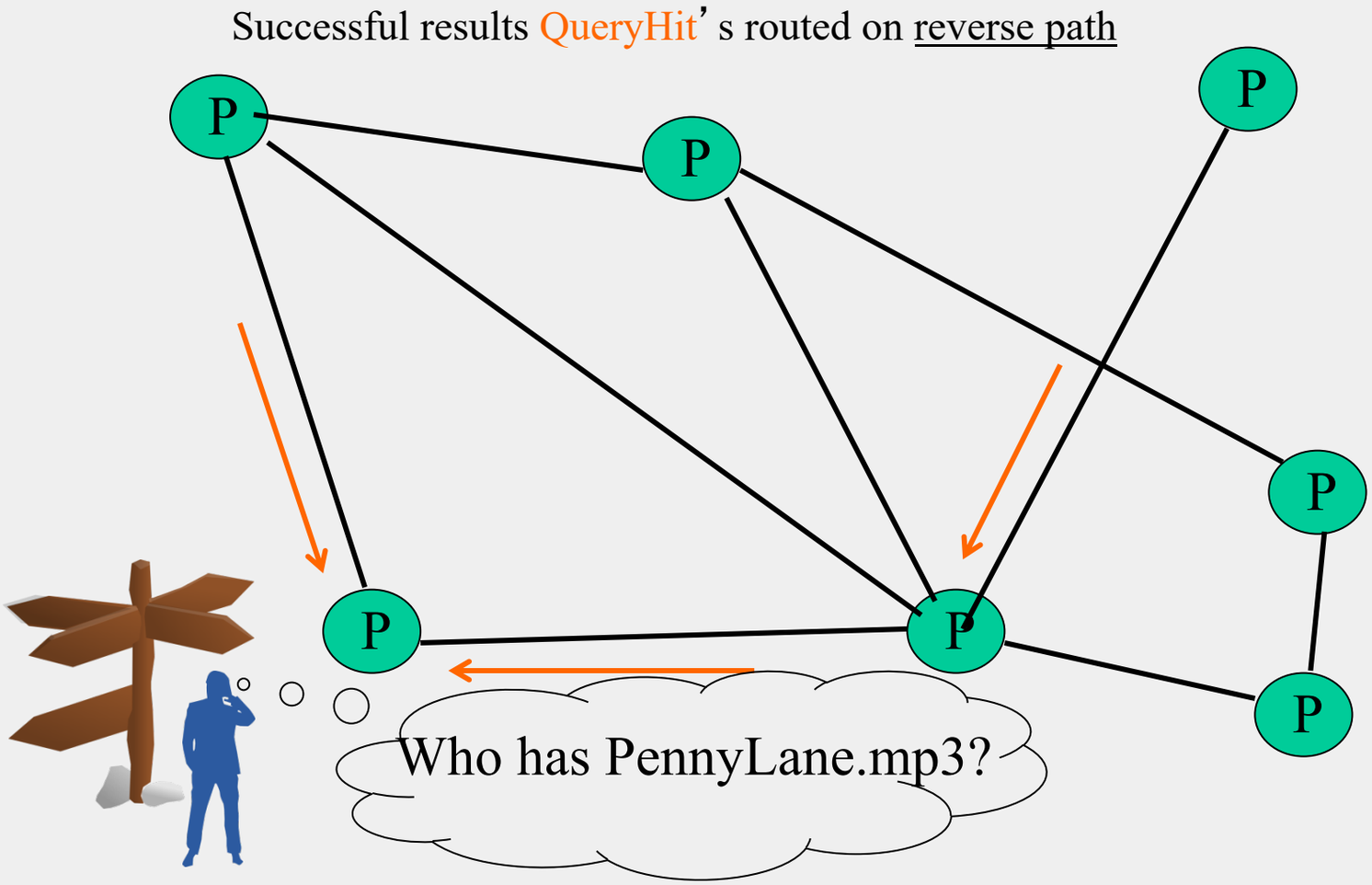
Gnutella Search

QueryHit (0x81) : successful result to a query



Payload Format in Gnutella QueryHit Message

Gnutella Search



Avoiding excessive traffic

- To avoid duplicate transmissions, each peer maintains a list of recently received messages
- Query forwarded to all neighbors except peer from which received
- Each Query (identified by DescriptorID) forwarded only once
- QueryHit routed back only to peer from which Query received with same DescriptorID
- Duplicates with same DescriptorID and Payload descriptor (msg type, e.g., Query) are dropped
- QueryHit with DescriptorID for which Query not seen is dropped

After receiving QueryHit messages

- Requestor chooses “best” QueryHit responder
  - Initiates HTTP request directly to responder’s ip+port

```
GET /get/<File Index>/<File Name>/HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0-\r\n
User-Agent: Gnutella\r\n
\r\n
```

- Responder then replies with file packets after this message:

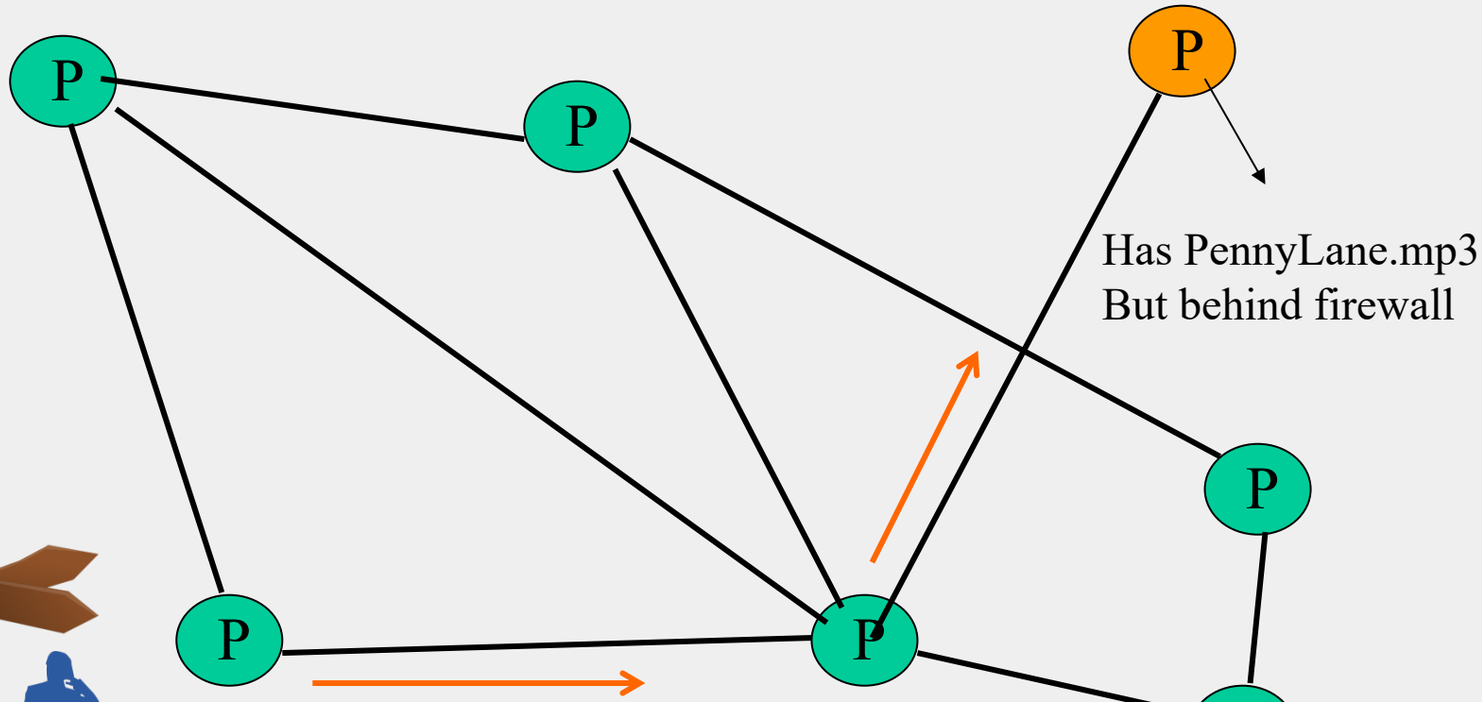
```
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type:application/binary\r\n
Content-length: 1024 \r\n
\r\n
```

After receiving QueryHit messages (2)

- HTTP is the file transfer protocol. Why?
  - Because it's standard, well-debugged, and widely used.
- Why the "range" field in the GET request?
  - To support partial file transfers.
- What if responder is behind firewall that disallows incoming connections?

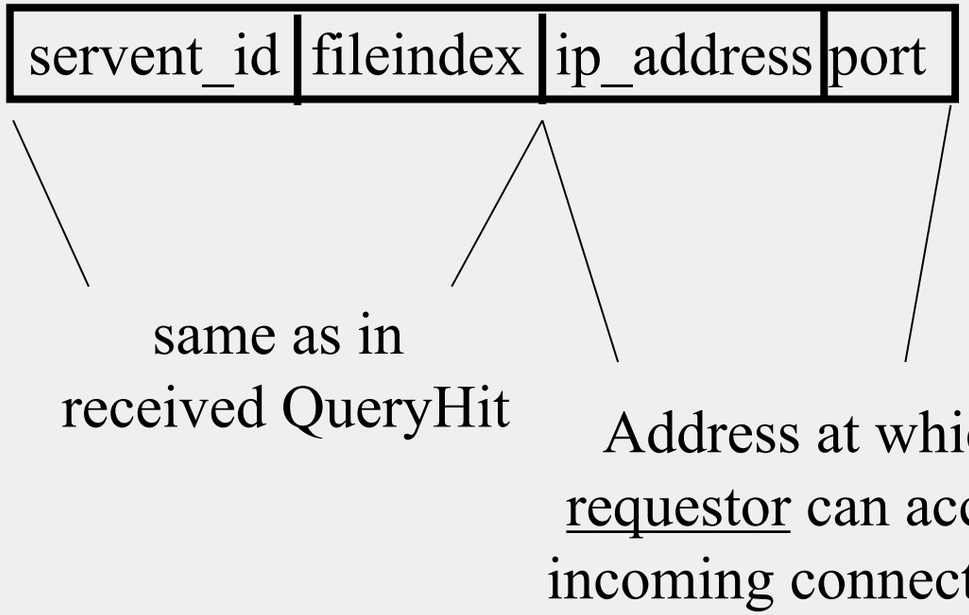
Dealing with Firewalls

Requestor sends Push to responder asking for file transfer



Dealing with Firewalls

Push (0x40)



Dealing with Firewalls

- Responder establishes a TCP connection at ip_address, port specified. Sends

  GIV <File Index>:<Servent Identifier>/<File Name>\n\n

- Requestor then sends GET to responder (as before) and file is transferred as explained earlier

- What if requestor is behind firewall too?
  - Gnutella gives up
  - Can you think of an alternative solution?

Ping-Pong

Ping (0x00)

no payload

Pong (0x01)

Port	ip_address	Num. files shared	Num. KB shared

- Peers initiate Ping's periodically
- Pings flooded out like Querys, Pongs routed along reverse path like QueryHits
- Pong replies used to update set of neighboring peers
  - to keep neighbor lists fresh in spite of peers joining, leaving and failing

Gnutella Summary

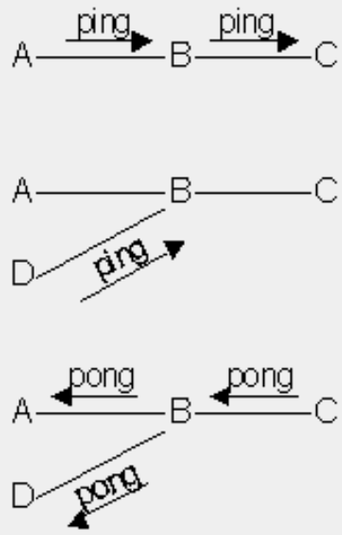
- No servers
- Peers/servents maintain “neighbors”, this forms an overlay graph
- Peers store their own files
- Queries flooded out, ttl restricted
- QueryHit (replies) reverse path routed
- Supports file transfer through firewalls
- Periodic Ping-pong to continuously refresh neighbor lists
  - List size specified by user at peer : heterogeneity means some peers may have more neighbors
  - Gnutella found to follow **power law** distribution:

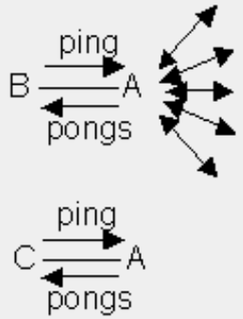
$$\text{P(\#links} = L) \sim L^{-k} \qquad (k \text{ is a constant})$$

Problems

- Ping/Pong constituted 50% traffic
  - Solution: Multiplex, *cache* and reduce frequency of pings/pongs
- Repeated searches with same keywords
  - Solution: *Cache* Query, QueryHit messages
- Modem-connected hosts do not have enough bandwidth for passing Gnutella traffic
  - Solution: use a central server to act as proxy for such peers
  - Another solution:

    ➔FastTrack System (soon)





Problems (contd.)

- Large number of *freeloaders*
  - 70% of users in 2000 were freeloaders
  - Only download files, never upload own files
- Flooding causes excessive traffic
  - Is there some way of maintaining meta-information about peers that leads to more intelligent routing?

    ➔ Structured Peer-to-peer systems

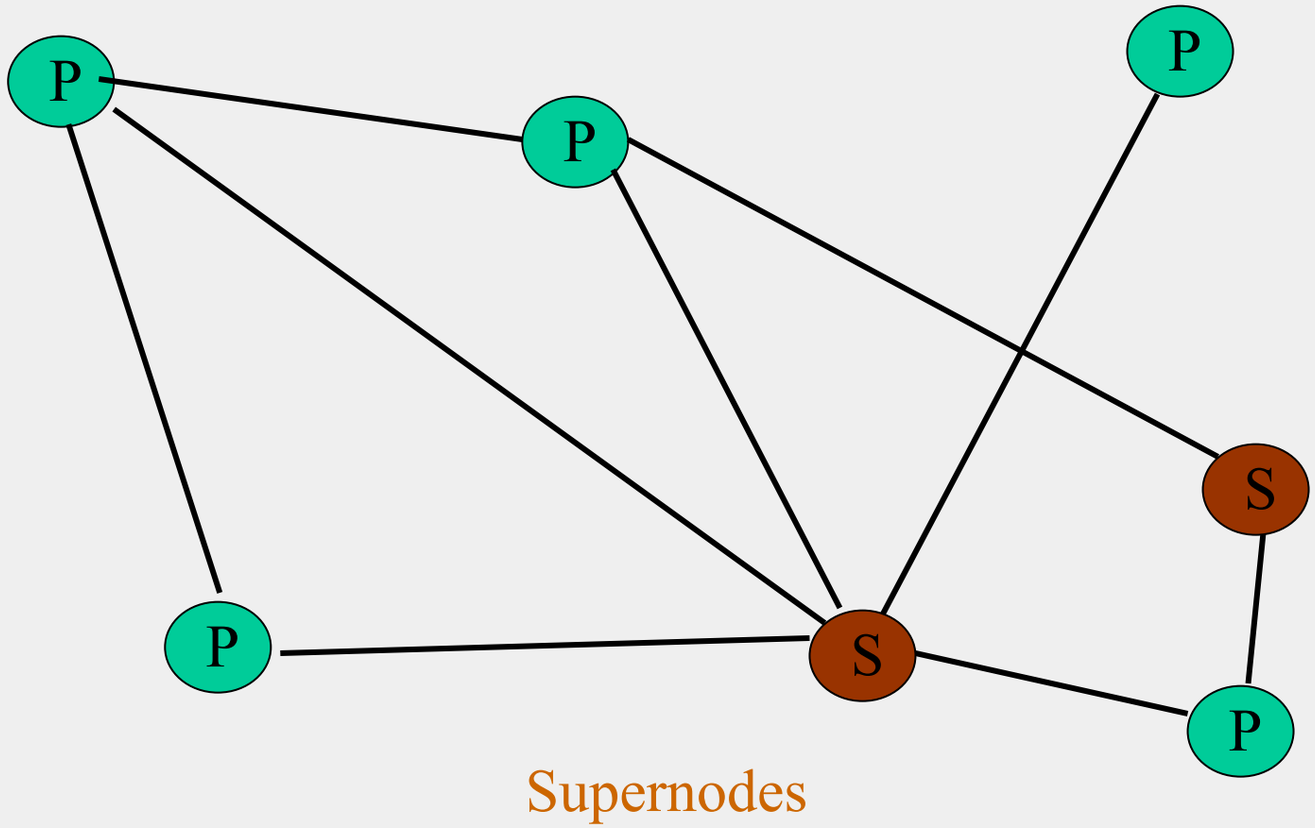
    e.g., Chord System (coming up next lecture)

FastTrack

- Hybrid between Gnutella and Napster
- Takes advantage of “healthier” participants in the system
- Underlying technology in Kazaa, KazaaLite, Grokster
- Proprietary protocol, but some details available
- Like Gnutella, but with some peers designated as *supernodes*

A FastTrack-like System

Peers



FastTrack (contd.)

- A supernode stores a directory listing a subset of nearby (<filename,peer pointer>), similar to Napster servers
- Supernode membership changes over time
- Any peer can become (and stay) a supernode, provided it has earned enough *reputation*
  - Kazaalite: participation level (=reputation) of a user between 0 and 1000, initially 10, then affected by length of periods of connectivity and total number of uploads
  - More sophisticated Reputation schemes invented, especially based on economics (See P2PEcon workshop)
- A peer searches by contacting a nearby supernode