COMP 4622: Computer Communication Networks II

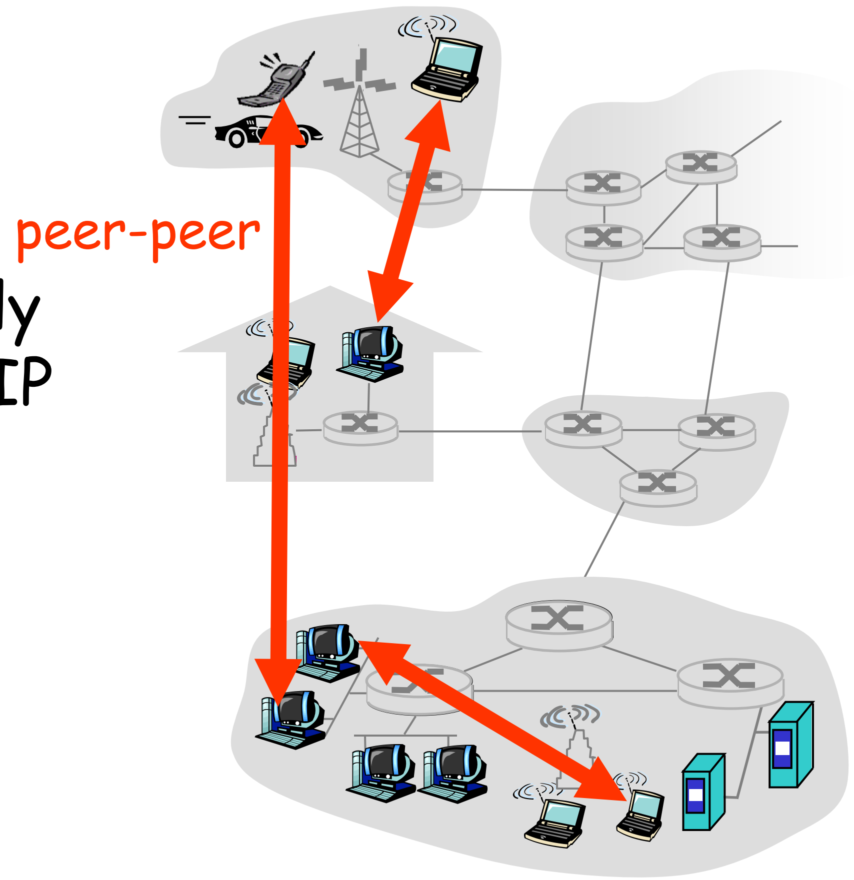
Peer-to-Peer Computing

Course Outline

- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
- Summary

Pure P2P architecture

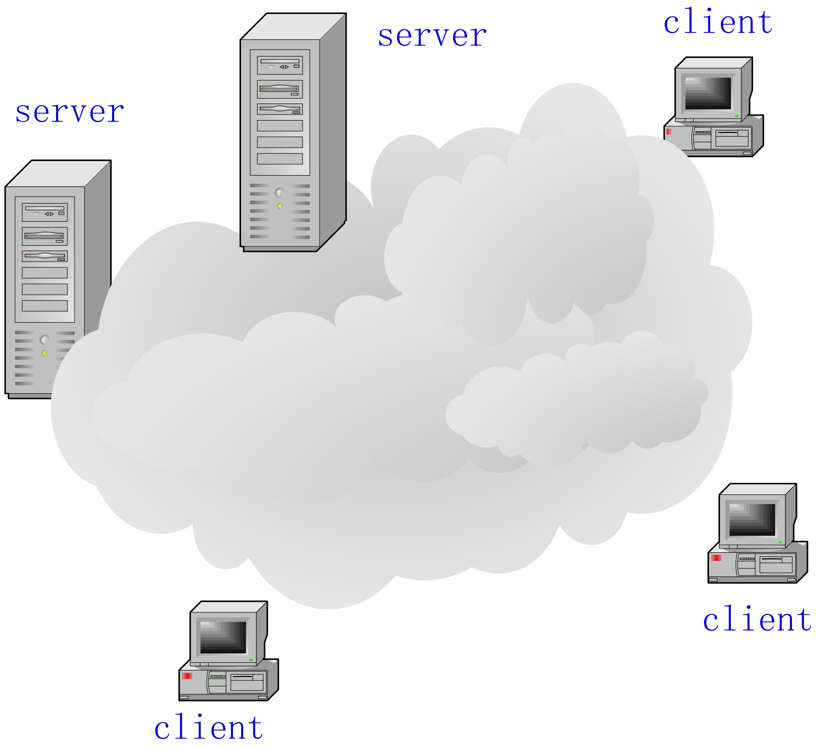
- *no* always-on server
- arbitrary end systems directly communicate
- peers are intermittently connected and change IP addresses



- Why P2P?
  - Compare to the client/server model

Client-Server Model

- Clients send request to servers
- Servers finish work and send back response to clients
- Small number of servers serve for a large number of clients
- Example: web service



Client/Server Model is Being Challenged!

- Client/server model seriously limits the utilization of available bandwidth and service
- Popular servers and search engines become traffic bottlenecks
- But high speed networks connecting many clients become idle
- Computing cycles, storage spaces and information in clients are under-utilized

Peer-to-Peer Model

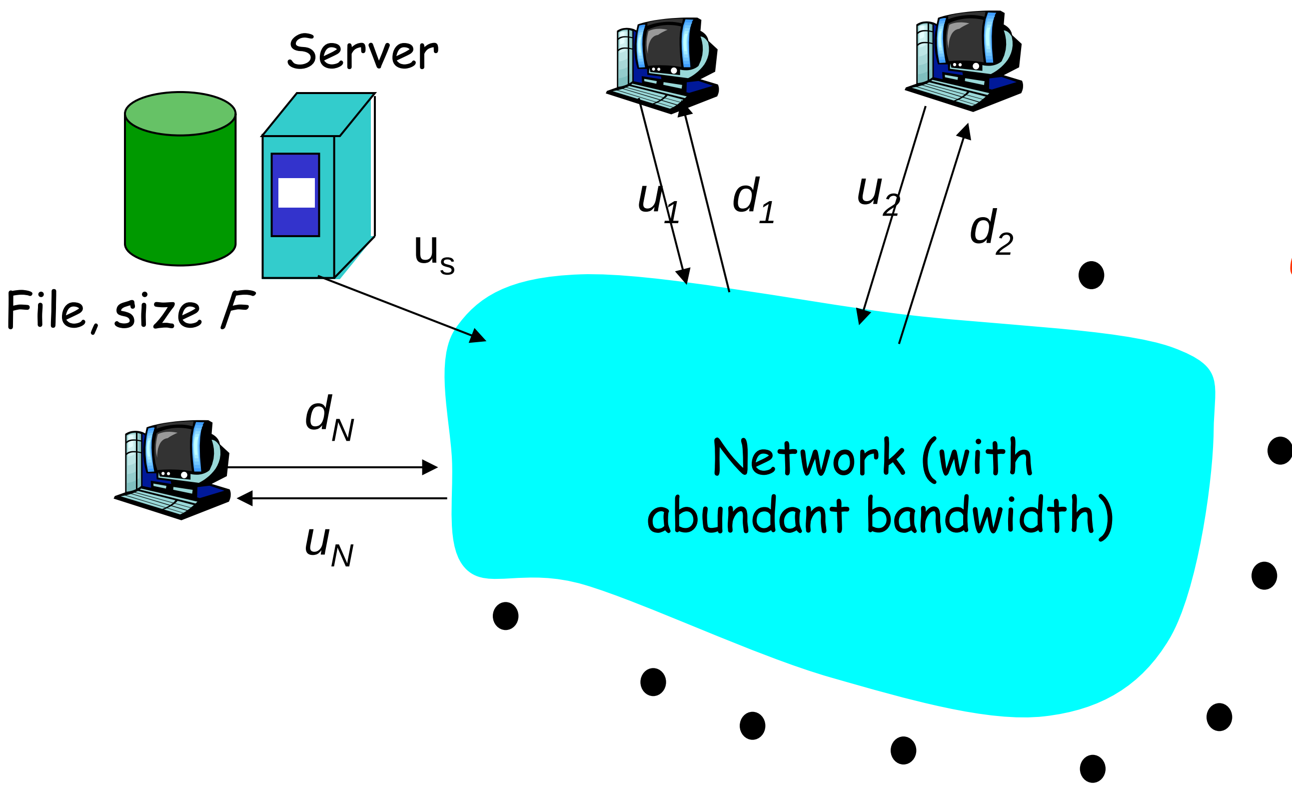
- P2P refers to a class of systems and applications that employ distributed resources to perform a critical function in a decentralized manner
  - Link the resources of all peers
  - Resources: storage, CPU cycles, content, etc.
  - All peers are equal and can serve requests – highly scalable
  - All peers are autonomous (different owners)
  - Peers are both clients and servers

P2P vs. C/S Models

- Equal peers vs. client/server relationship
  - Scalability
- Dynamic peers vs. fixed servers
  - Dynamic joining and leaving
  - Intermittent connectivity
  - Locating peers (unknown IP) and servers (known IP)!!!
- Complement to each other
  - Application dependent

File Distribution: Server-Client vs. P2P

Question : How much time to distribute file from one server to N *peers*?

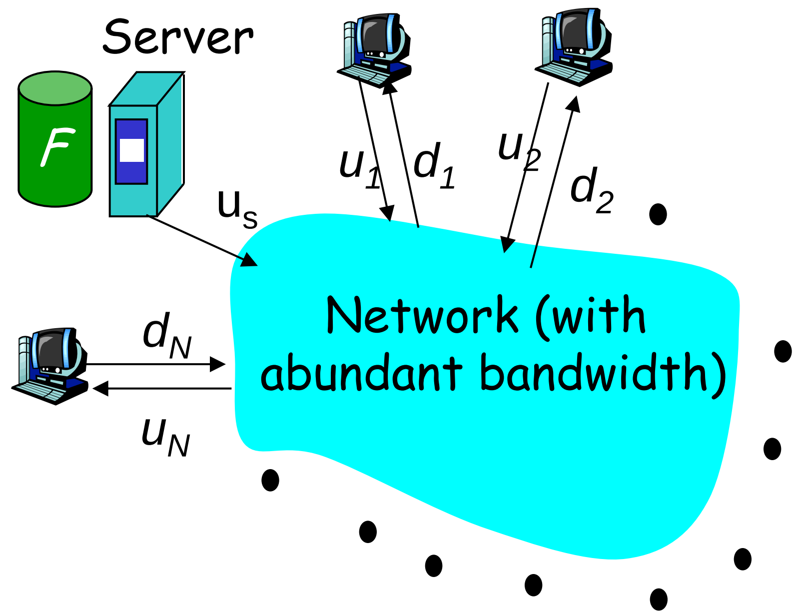


u_s: server upload bandwidth

u_i: peer i upload bandwidth

d_i: peer i download bandwidth

File distribution time: server-client

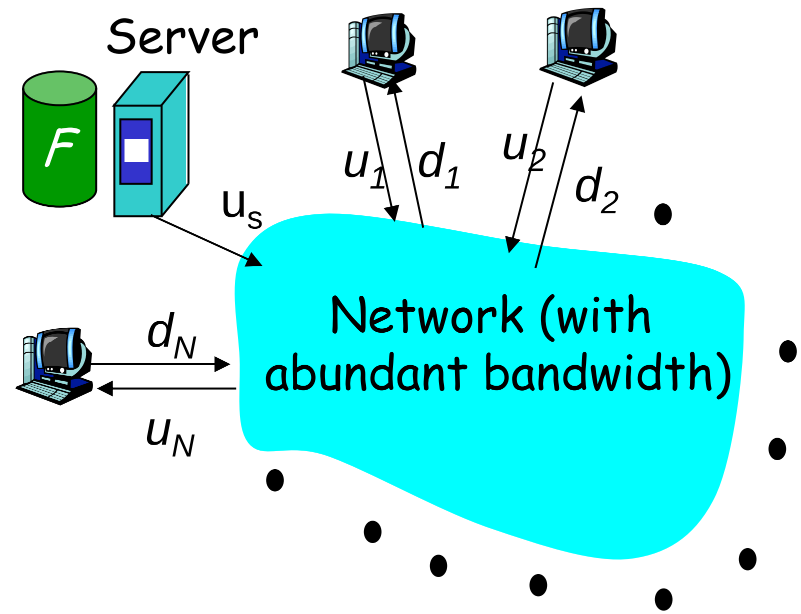


- server sequentially sends N copies:
  - NF/u_s time
- client i takes F/d_i time to download

Time to distribute F to N clients using client/server approach $= d_{cs} = \max\{NF/u_s, F/\min_i(d_i)\}$

increases linearly in N (for large N)

File distribution time: P2P



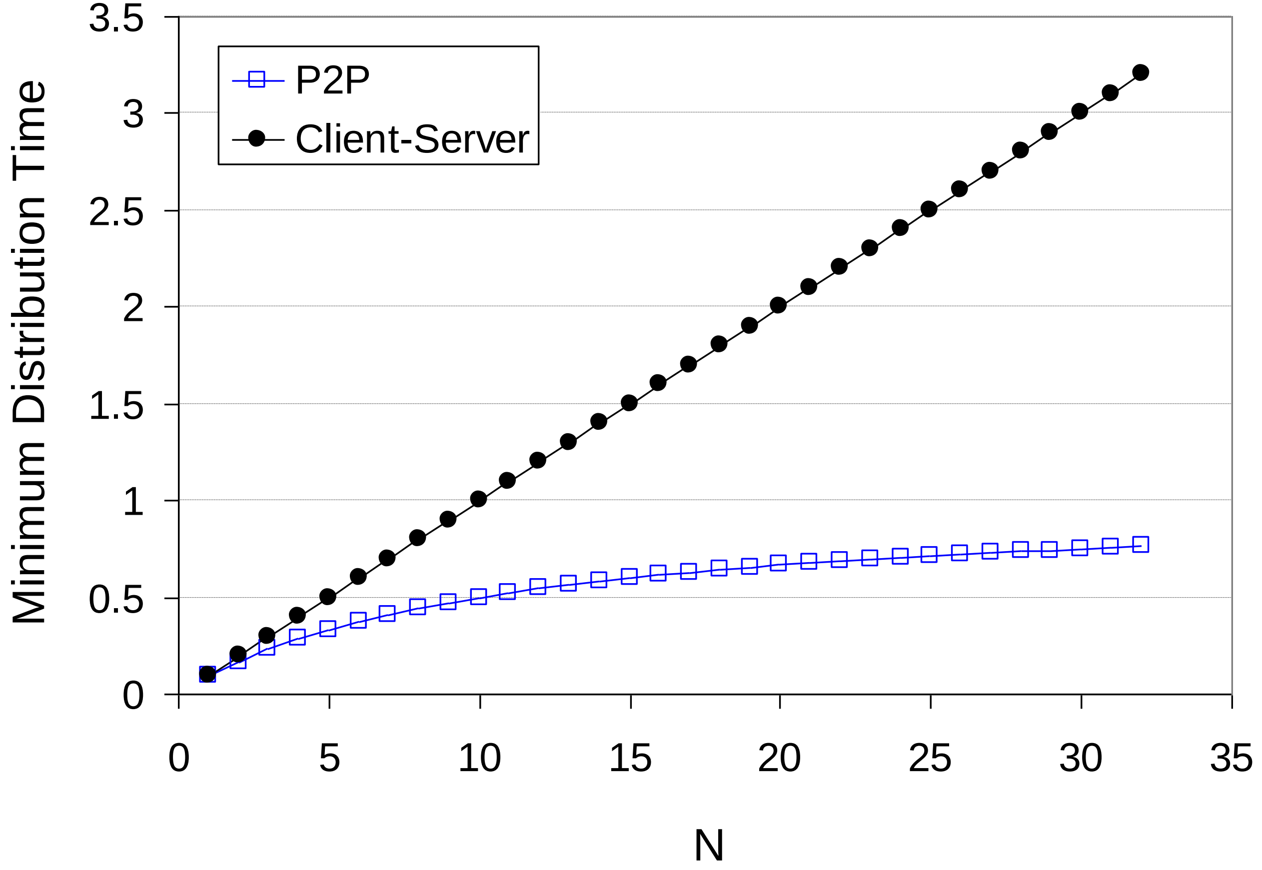
- server must send one copy: F/u_s time
- client i takes F/d_i time to download
- NF bits must be downloaded (aggregate)

- fastest possible upload rate: $u_s + \Sigma u_i$

$$d_{P2P} = \max\{\,F/u_s,\ F/\min(d_i)\,,\ NF/(u_s + \Sigma u_i)\,\}$$

Server-client vs. P2P: example

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



Course Outline

- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
- Summary

History of P2P

- Some P2P applications
  - File sharing: most popular
    - Napster, KaZaA, Gnutella, etc.
  - P2P Communication
    - Instant messaging: Yahoo, AOL, MSN
  - P2P Computation
    - seti@home
    - Grid computing
- Napster played a major role in promoting P2P

History of Napster

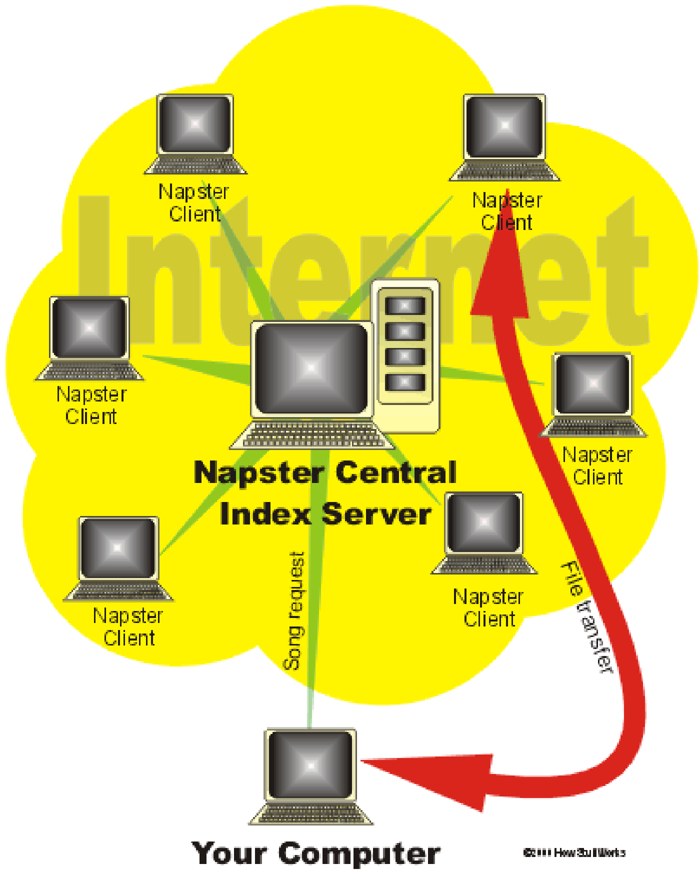
- 5/99: Shawn Fanning (freshman, Northeastern University) founded Napster Online music service
- 12/99: first law suit from Recording Industry Association of America (RIAA)
- Central index server has the metadata of some pirated MP3s
- 3/00: 25% UWisc traffic Napster

History of Napster

- 2/01: US Circuit Court of Appeals: Napster knew users violating copyright laws
- 7/01: stopped operation (failed to become a pay-based service); 160K simultaneous online users
- Triggered other P2P file sharing applications and research in P2P

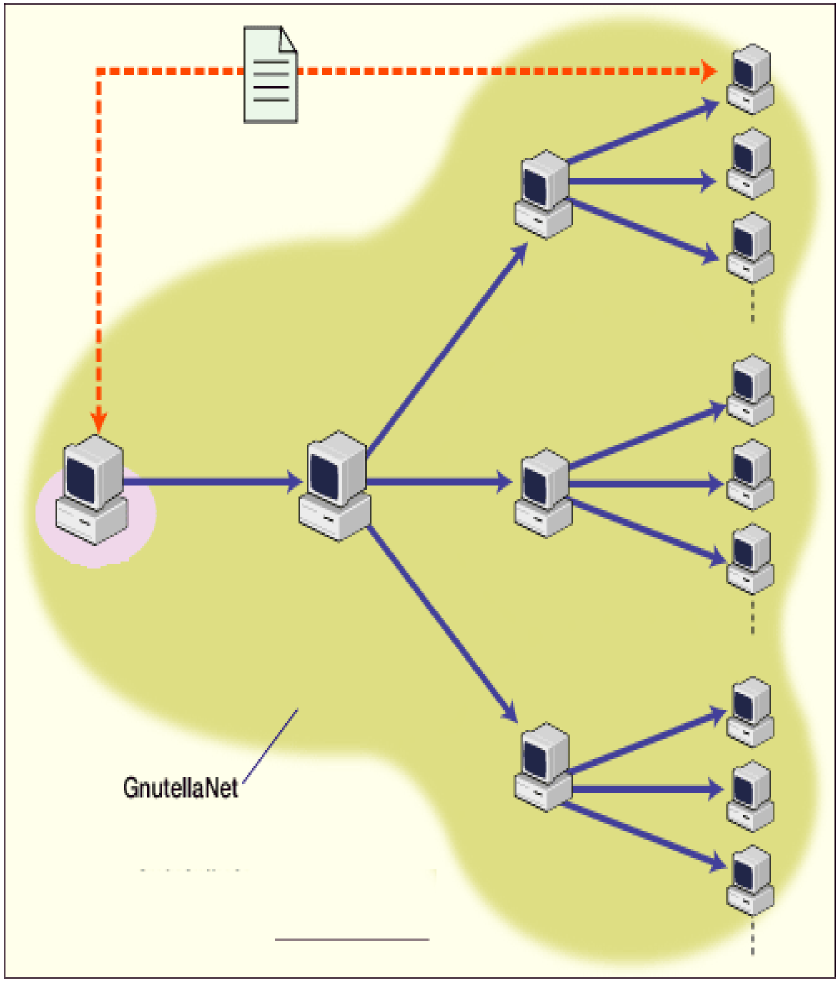
Napster

- *Sharing* of MP3 files
- Central server caches information
- Query is 'reliable'
- Does not 'scale'
- Opens again early 2002 (with membership)



Gnutella

- Re-engineered after AOL shutdown
- Decentralized
- Flooding/broadcasting
- Due to TTL, query covers less than 100%
- approx. 40.000 online
- Power-Law connections



P2P impact today (1)

- Widespread adoption
  - KaZaA – 360 million downloads (1.3M/week) one of the most popular applications ever!
- leading to (almost) zero-cost content distribution:

  … is forcing companies to change their business models

  … might impact copyright laws

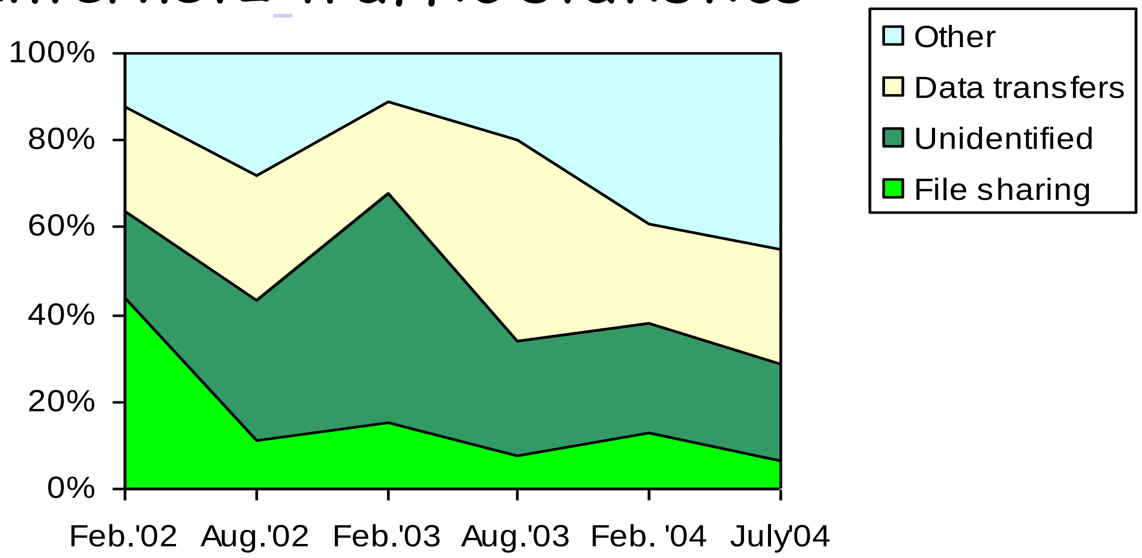
FastTrack	2,460,120
eDonkey	1,987,097
Overnet	1,261,568
iMesh	803,420
Warez	440,289
Gnutella	389,678
MP2P	267,251

Sources: www.slyck.com, www.kazaa.com, July ‘04

P2P impact today (2)

- P2P - file-sharing - generated traffic may be the single largest contributor to Internet traffic today
- Driving adoption of consumer broadband

Internet2 traffic statistics



Source: www.internet2.edu, July ‘04

P2P impact today (3)

- A huge pool of underutilized resources lays around,
- users are willing to donate these resources
- which can be put to work efficiently (at least for some types of applications)

	Total	Last 24 hours
Users	4,236,090	23,365
Results received	764M	1.13M
Total CPU Time	1.3M years	1.3K years
Floating point operations		**51.4 TFLOPS**

Source: Seti@Home website, Oct. 2003

Some Popular P2P Systems

More Examples

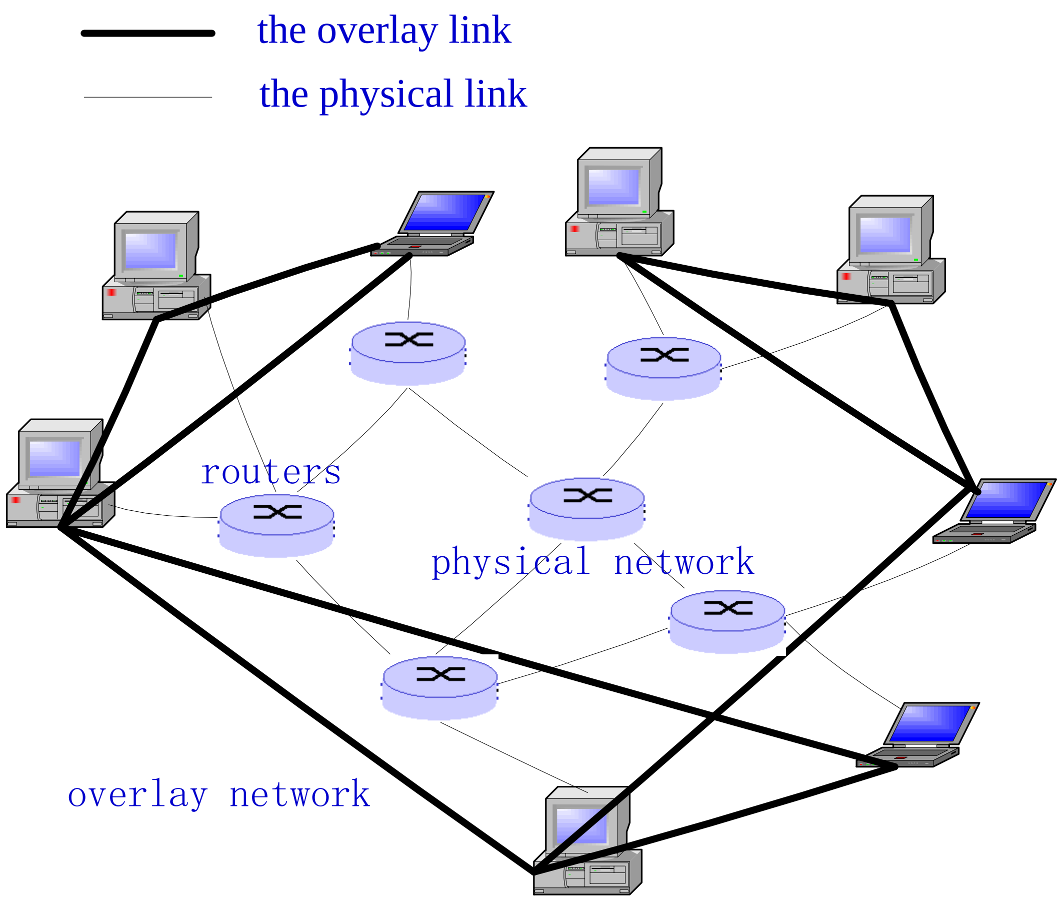
- Seti@home
- Napster
- Gnutella
- Freenet
- Morpeus
- Yahoo! Messenger
- MSN
- ICQ
- Doom/quake
- Jxta
- …

Course Outline

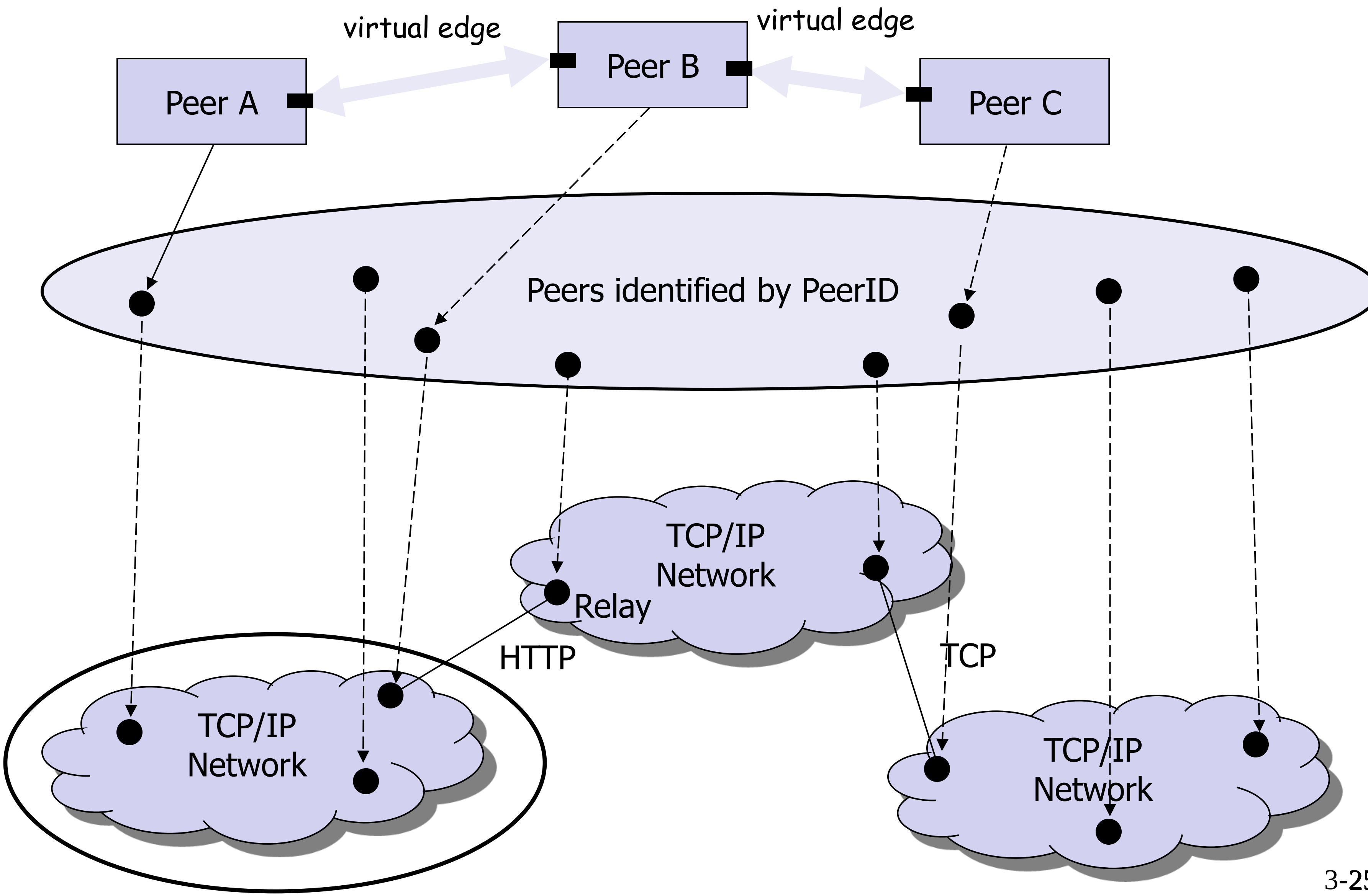
- Why P2P?
- A brief history
- P2P architectures
  - P2P overlay networks
  - Centralized and decentralized
  - Structured and unstructured
- Usage of P2P technology
- Summary

Overlay Networks (I)

- Application layer
- Flexible design
  - Topology
  - Maintenance
  - Protocol
  - Messaging over TCP or UDP
- Transparent to the underlying IP network
- Disadvantages
  - Longer latency
  - Setup and maintenance traffic



Overlay Networks (II)



P2P Overlay Networks

- Peers are nodes
- Two neighboring peers are connected by a virtual edge
  - TCP connection
  - Pointer to an IP address
- Overlay maintenance (changing topology)
  - New node needs to bootstrap
  - Ping neighbors periodically
  - Verify liveness while messaging
  - Establish new neighbors if necessary

Some Terminologies

- Query and responses
- Partial/exact match
- Full/partial coverage
  - Full: query will reach all peers within a specific area
- Popular/rare files
  - Frequency of files being searched and downloaded
- Scope: ability to find both popular and rare files
- Anonymity
  - Provider anonymity
  - Requester anonymity
  - Creator anonymity

Design Considerations

- Overlay construction & maintenance
    - Overlay topology, select neighbors, number of neighbors, replace neighbors
- Search characteristics
    - Anonymity, support partial/exact match, full/partial coverage
- Search quality
    - Scope
- Search performance
    - Search traffic and response time
    - Selection of responses
- Parallel download from multiple peers

Taxonomy of P2P File Sharing Networks

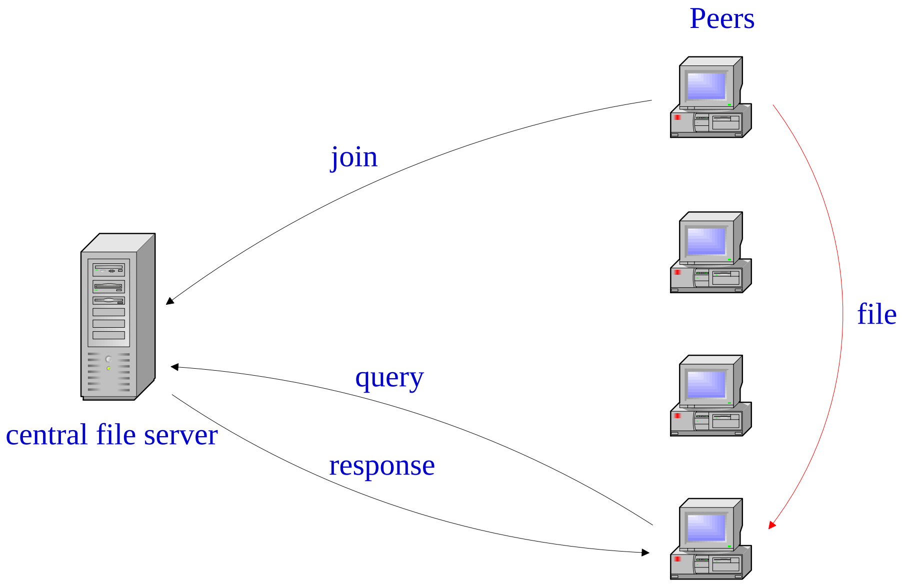
- Unstructured: file placement is unrelated to the overlay topology
  - With a central server: Napster
  - Fully decentralized: Gnutella & Freenet
  - Hierarchical: Kazaa
- Structured: the overlay topology & file (or file indices) placement are tightly controlled
  - One-dimensional coordinate space: Chord

Course Outline

- Why P2P?
- A brief history
- P2P architectures
    - P2P overlay networks
    - Centralized and decentralized
    - Structured and unstructured
- Usage of P2P technology
- Summary

Napster

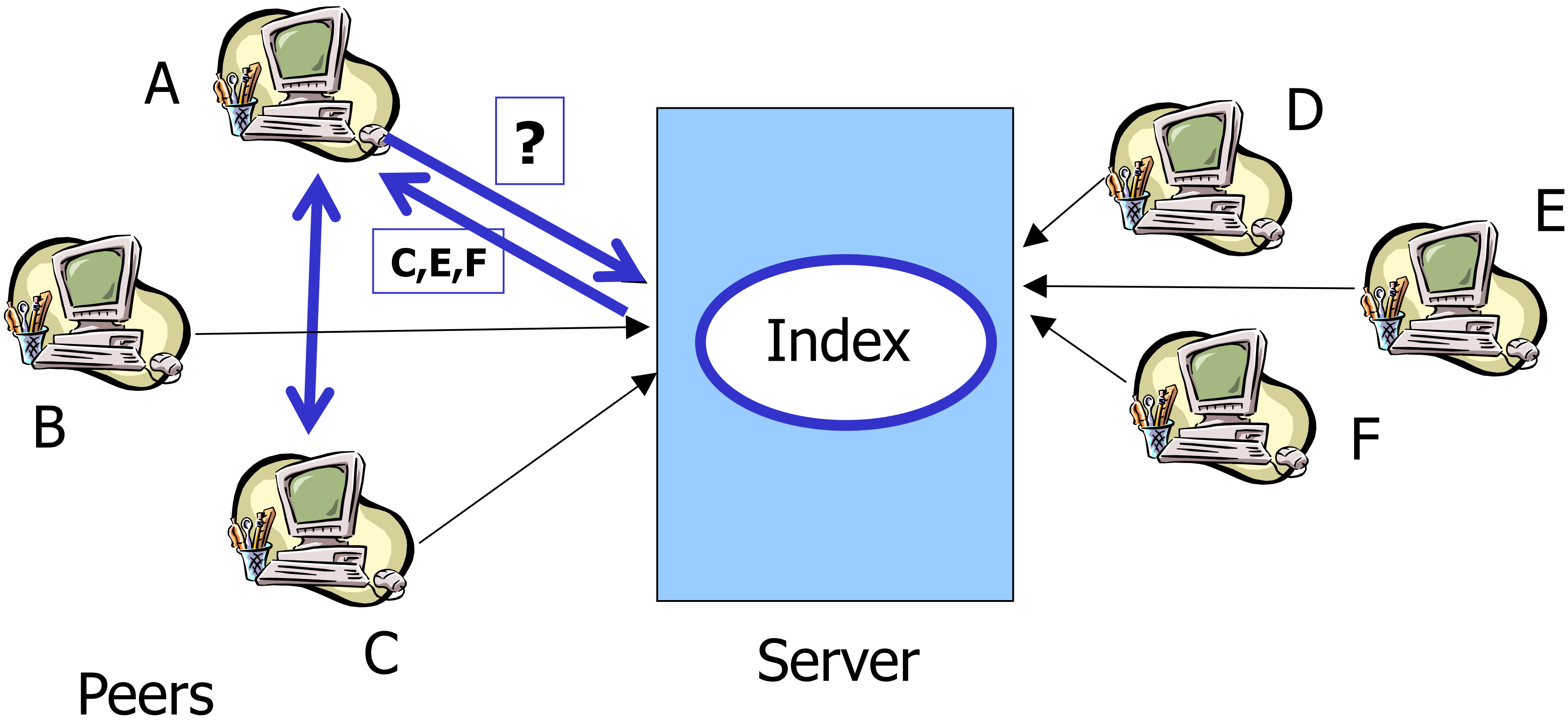
- Centralized directory server
- Search
  - Connect to Napster server, upload MP3 indices
  - Give server keyword to search
  - Select host/file to download (ping hosts to select from)
  - Directly download from the selected peer



- Efficient because centralized
- Disadvantage
  - Single point of failure
  - Poor anonymity
  - Performance bottleneck

Search in Napster

- "Centralized" P2P system

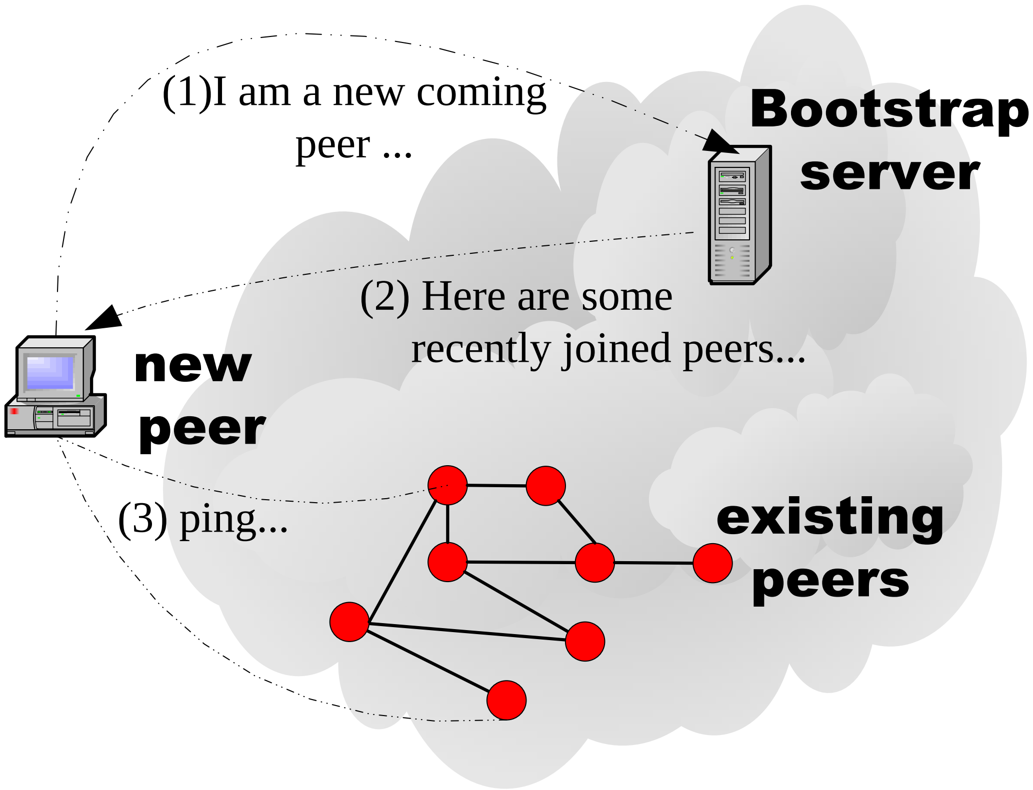


Gnutella

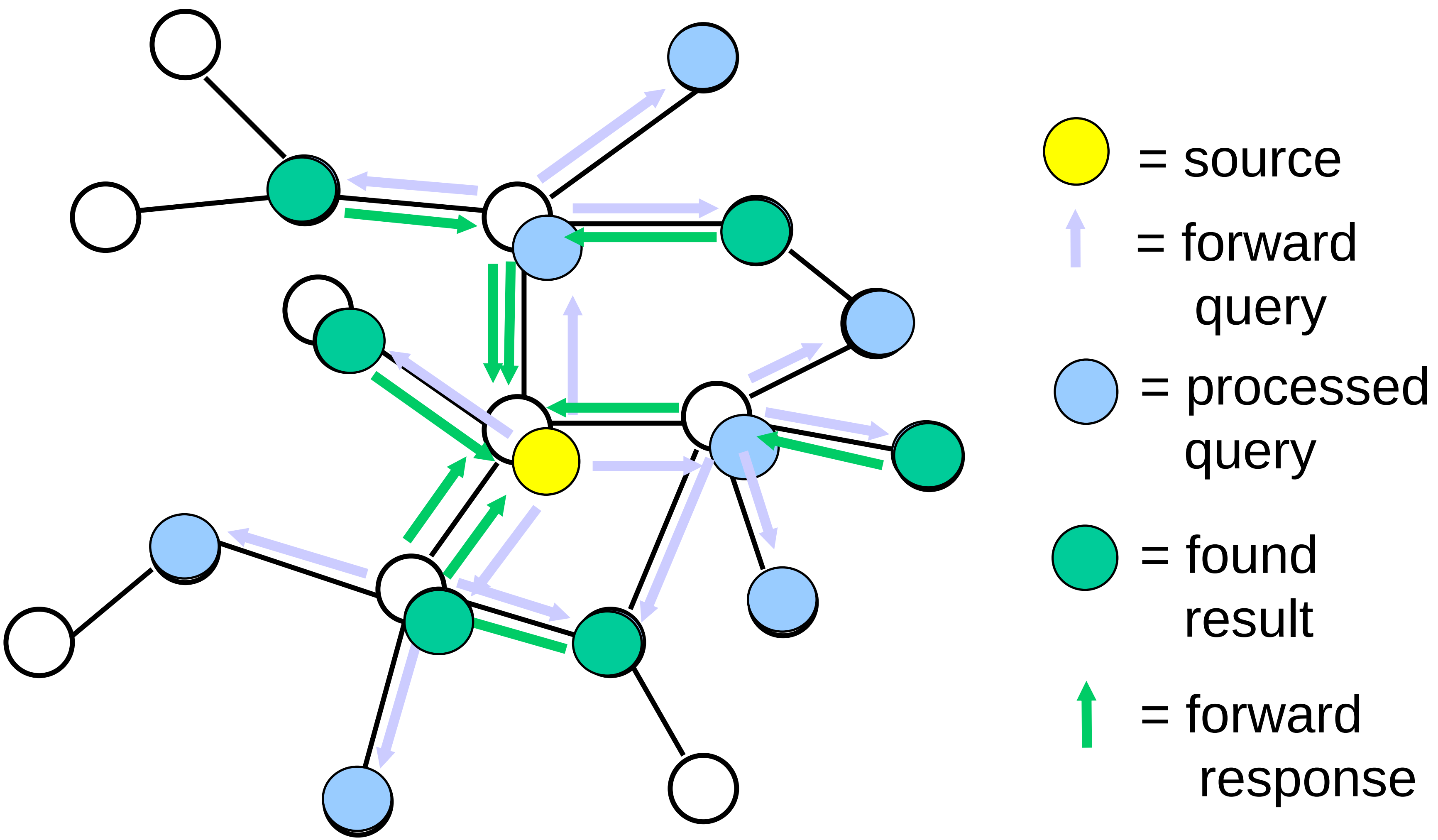
- History
  - Developed by Justin Frankel and Tom Pepper at Nullsoft (subsidiary of AOL)
  - 3/14/00: released by AOL, withdrawn immediately
  - Open source
  - Many implementations (e.g., limewire)
  - Still try to improve
- Fully decentralized
  - Difficult to pull plug

Gnutella: Bootstrapping

- Some well known bootstrapping servers keep a list of some peering nodes
- When a new peer wants to join a P2P network, it contacts a bootstrapping server, the IP addresses of a list of existing peers are provided.
- The new peer then tries to connect with some of these peers.



Search in Gnutella

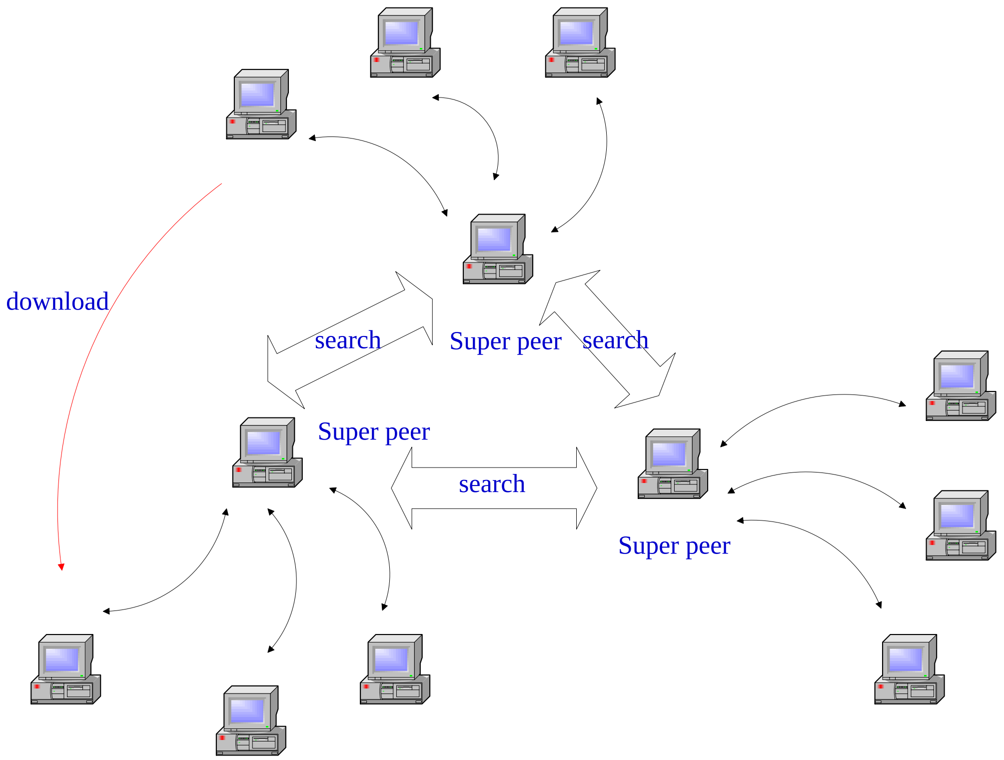


Gnutella

- Benefits
  - No server needed (cost)
  - Robust (nodes can come and go)
  - Can handle complex queries per node
- Disadvantages
  - Not comprehensive (can miss results)
  - Inefficient! (many messages)

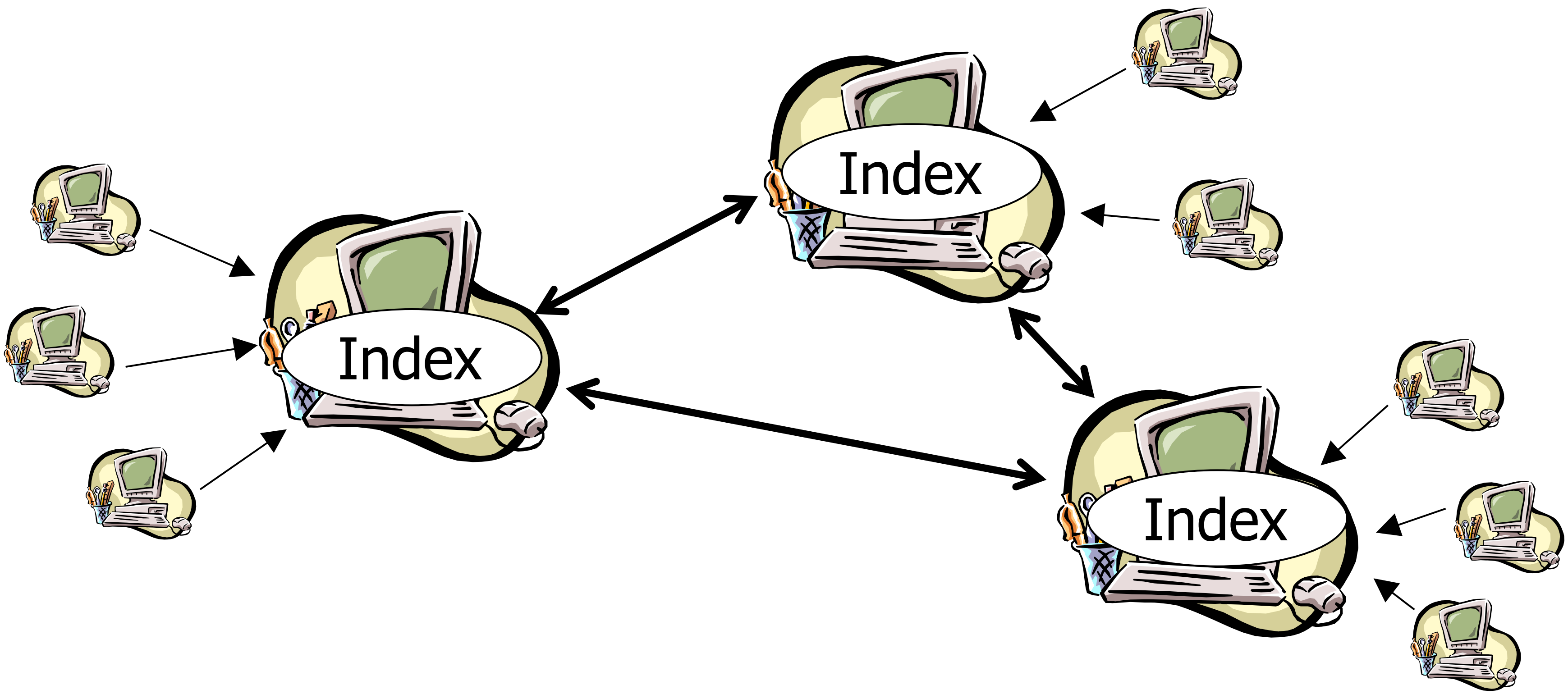
KaZaA: Hierarchical Structure

- Hybrid architecture
  - A decentralized network of centralized clusters
  - Sacrifice anonymity to achieve efficiency
  - Cross between Napster and Gnutella
- Each node is either a supernode or assigned to a supernode
- Each supernode knows about many other supernodes (almost mesh overlay)



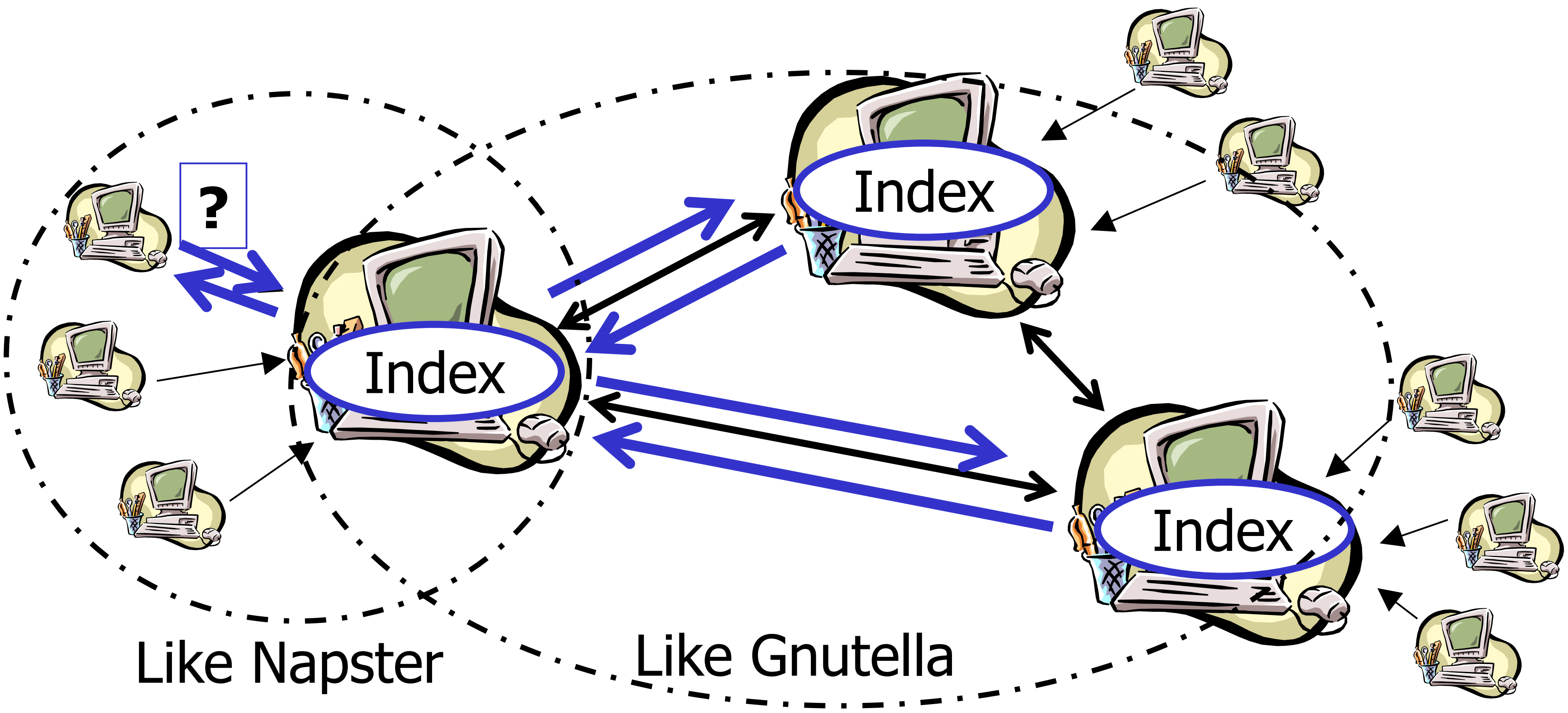
KaZaA

- "Super-peer" P2P system



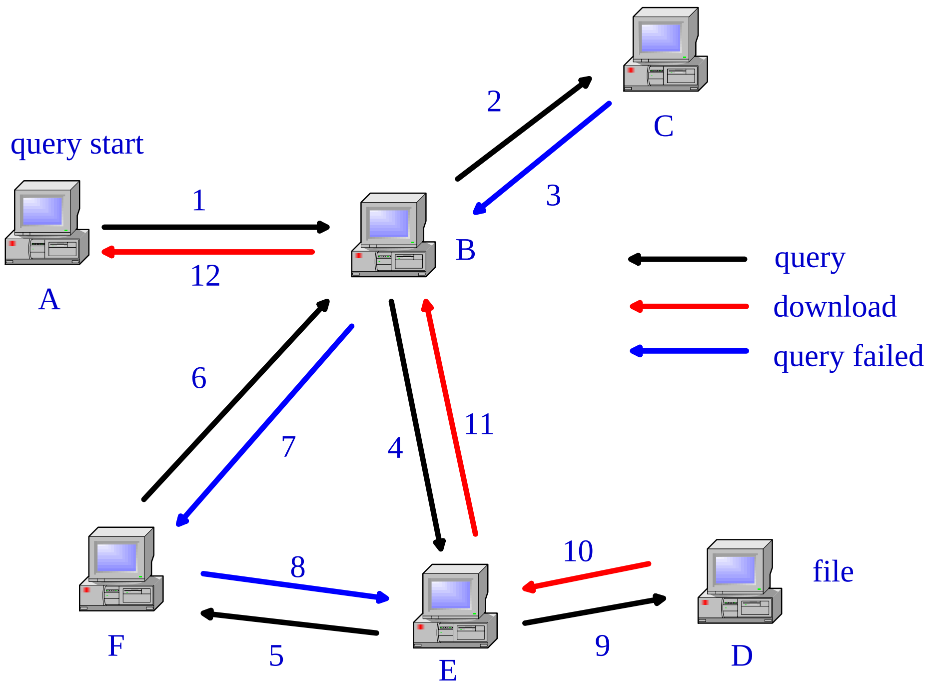
Search in KaZaA

- "Super-peer" P2P system



Search in Freenet

- Decentralized
- Search
  - Depth first search (chain mode search)
  - Peers have routing tables: neighboring peers and their files' GUID
  - Peers forwards the queries to the peer in their table with the closest GUID
  - File sent back along the request path



- Disadvantage
  - Long response time

Course Outline

- Why P2P?
- A brief history
- P2P architectures
  - P2P overlay networks
  - Centralized and decentralized
  - Structured and unstructured
- Usage of P2P technology
- Summary

Structured P2P Systems

- A number of contenders
  - Chord [SIGCOMM'01], Pastry [Middleware'01], Tapestry [SPAA'02], Kademlia [IPTPS'02]
- DHT: Distributed Hash Tables

Distributed Hash Table (DHT)

- A large shared memory (database) implemented by p2p nodes
- Addresses are logical, not physical
- Implies applications can select them as desired
- Typically a hash of other information
- System looks them up

Distributed Hash Table (DHT)

- DHT implements a distributed P2P database
- Database has (key, value) pairs;
  - key: ss number; value: human name
  - key: content type; value: IP address
- Peers query DB with key
  - DB returns values that match the key
- Peers can also insert (key, value) peers

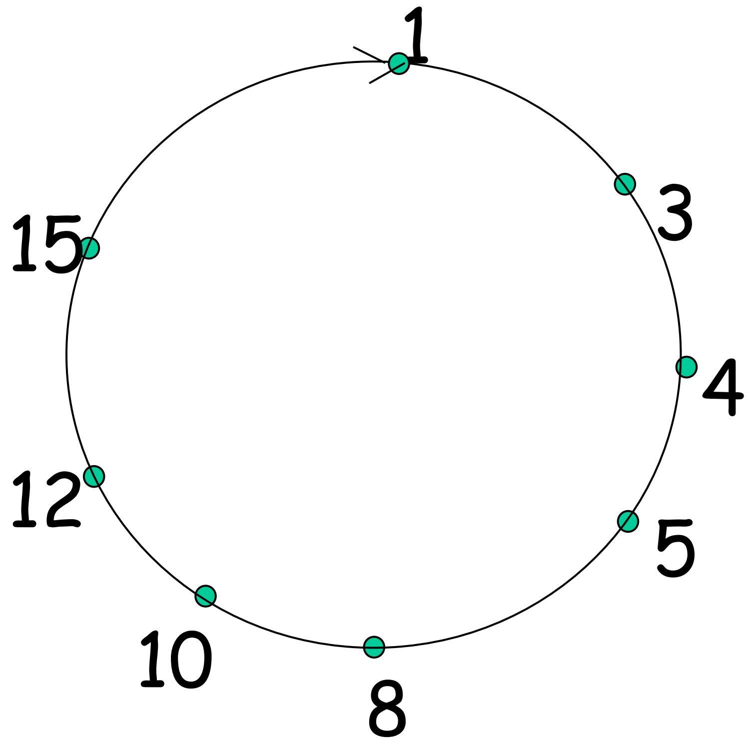
DHT Identifiers

- Assign integer identifier to each peer in range $[0,2^n-1]$.
  - Each identifier can be represented by n bits.
- Require each key to be an integer in same range.
- To get integer keys, hash original key.
  - eg, key = h("Led Zeppelin IV")
  - This is why they call it a distributed "hash" table

How to assign keys to peers?

- Central issue:
  - Assigning (key, value) pairs to peers.
- Rule: assign key to the peer that has the closest ID.
- Convention in lecture: closest is the immediate successor of the key.
- Ex: n=4; peers: 1,3,4,5,8,10,12,14;
  - key = 13, then successor peer = 14
  - key = 15, then successor peer = 1

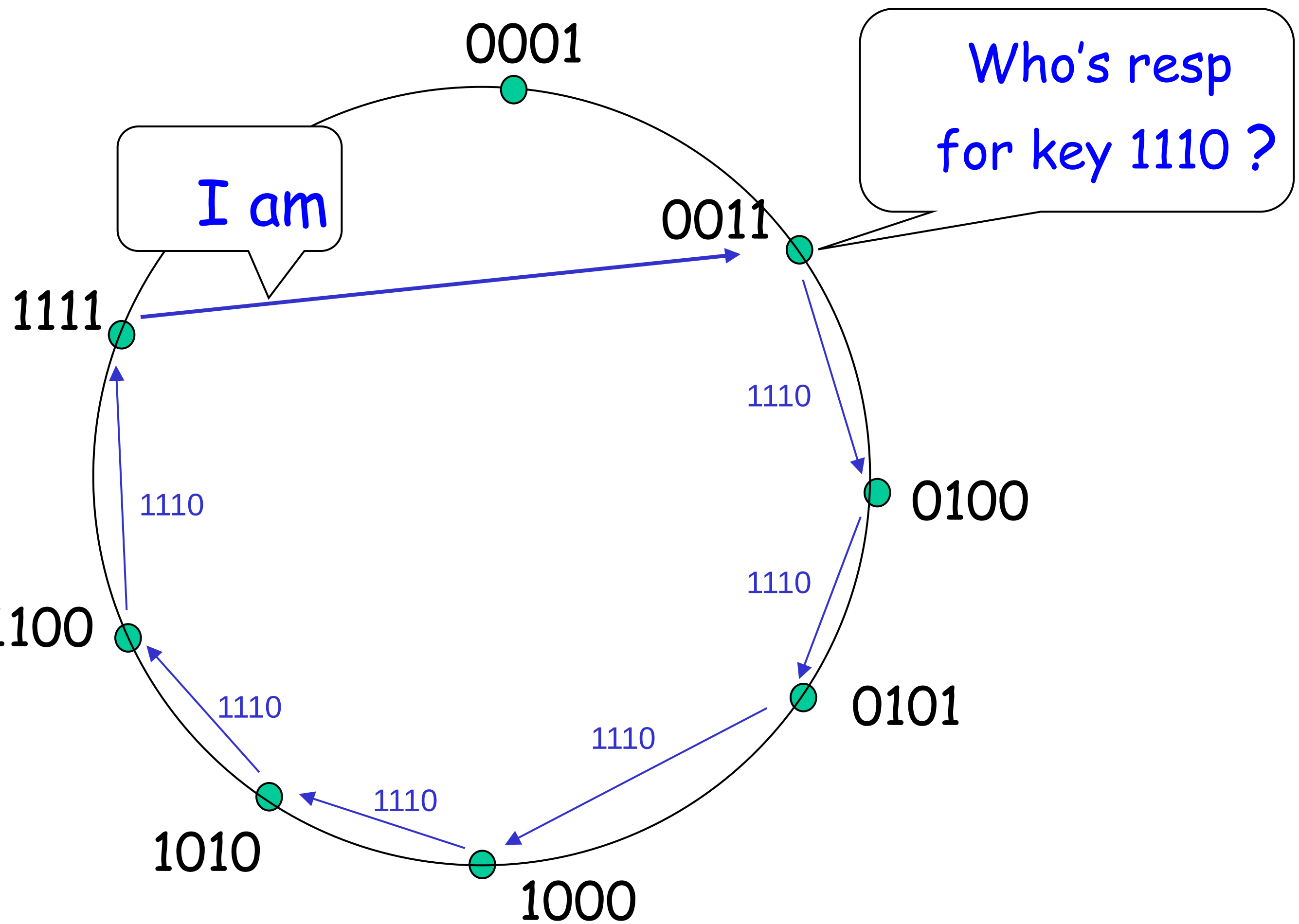
Circular DHT (1)



- Each peer *only* aware of immediate successor and predecessor.
- "Overlay network"

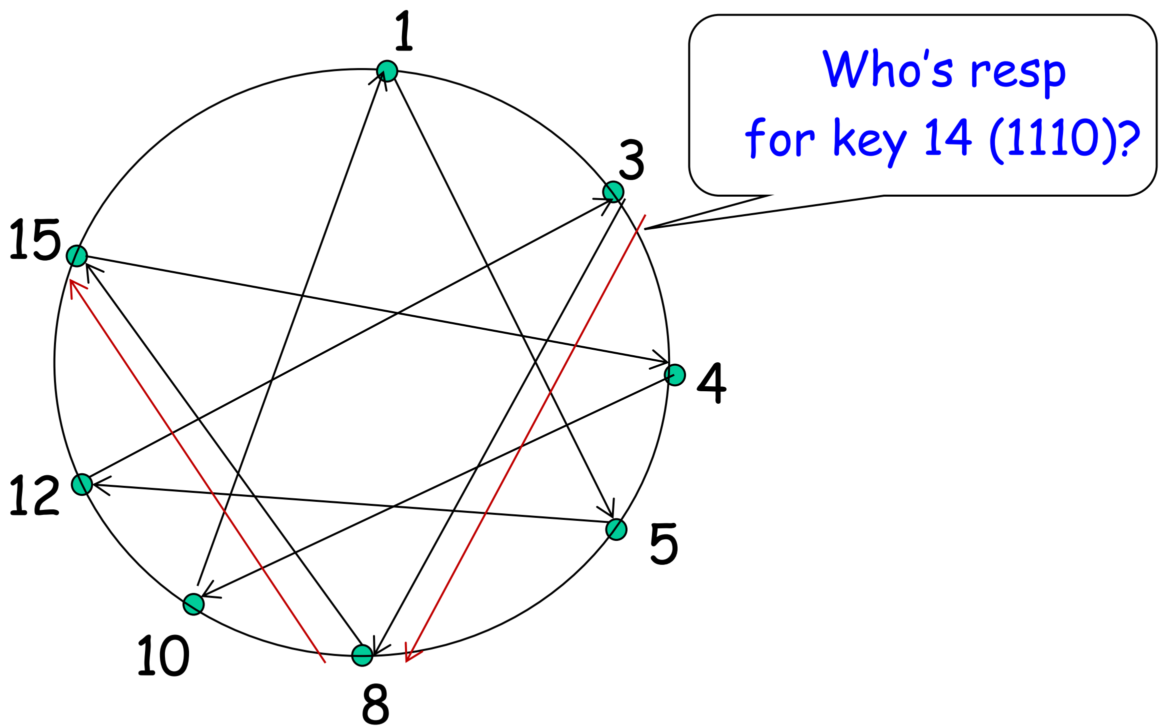
Circular DHT (2)

O(N) messages
on avg to resolve
query, when there
are N peers



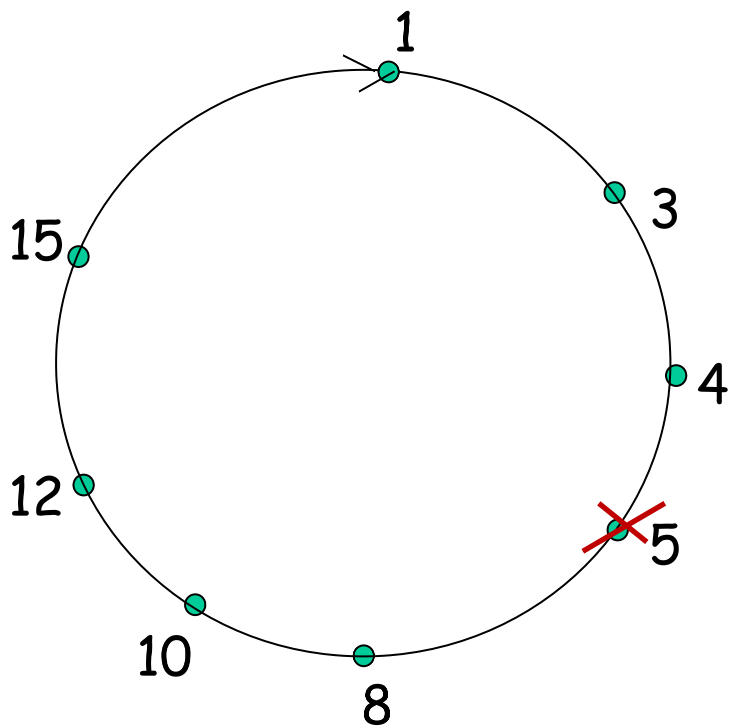
Define closest
as closest
successor

Circular DHT with Shortcuts



- Each peer keeps track of IP addresses of predecessor, successor, short cuts.
- Reduced from 6 to 2 messages.
- Possible to design shortcuts so O(log N) neighbors, O(log N) messages in query

Peer Churn

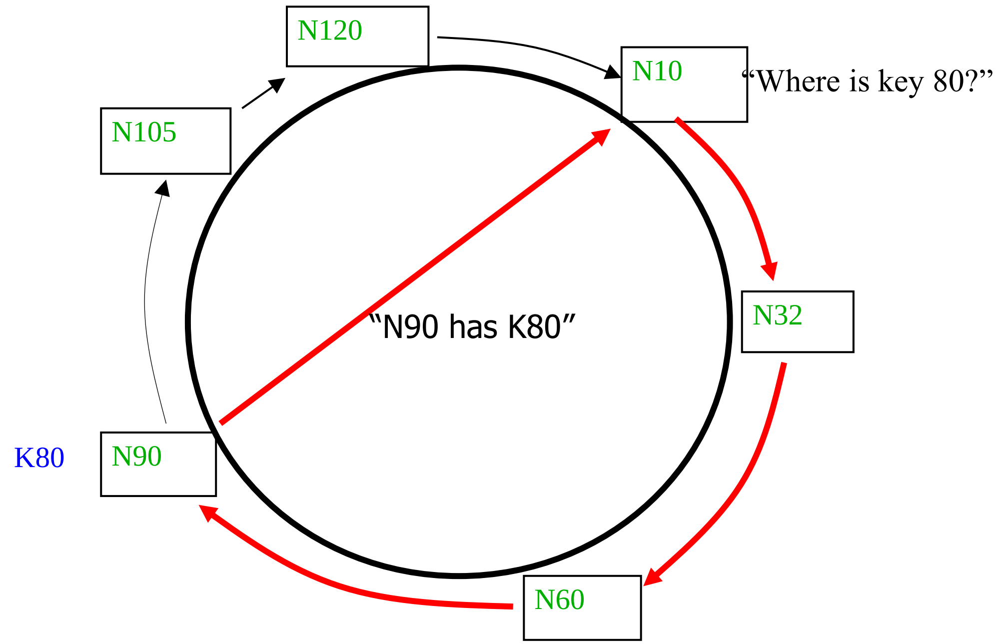


- To handle peer churn, require each peer to know the IP address of its two successors
- Each peer periodically pings its two successors to see if they are still alive

- Peer 5 abruptly leaves
- Peer 4 detects; makes 8 its immediate successor; asks 8 who its immediate successor is; makes 8's immediate successor its second successor.
- What if peer 13 wants to join?

Search in Chord

- Topology:
  - Ring, like token ring network
- Search:
  - Each peer is assigned a key by DHT
  - Each file is assigned a key by DHT
  - Files/file indices are kept in the first peer with a larger key
  - Search is routed along the ring
  - Sacrifice autonomy to achieve efficiency



- Disadvantage:
  - Cannot support partial match
  - Much maintenance traffic

Course Outline

- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
  - Improvement of search strategies
  - Topology mismatch problem
  - Trustworthy computing
- Summary

Why we care about Search in Unstructured P2P Networks?

- Unstructured networks are most flexible, practical and widely deployed
- Search mechanism is a major performance factor of unstructured P2P networks
- Searching mechanisms used now are not efficient, cause too much traffic
- Locate desired data in resource-efficient manner
  - Traffic cost
  - Response time
  - Success rate…

General Search Mechanisms

- BFS
  - Able to find the “nearest” peers
- DFS
  - Might dive right down to find a deep target peer
  - Might head down one branch of the search tree and never return

Heuristic Search

- BFS and DFS
  - “Blind" or "knowledge-free" search techniques
- Heuristic search
  - Use information to improve the performance
  - In P2P networks, historical or aggregated file location information may be used

How to Improve These Criteria

- Reduce the search traffic on peers & Internet
  - Decrease the number of query messages
- Shorten response time
  - Decrease the number of hops and the delay of each hop
- Find rare files
  - Replicate rare files or file indices
  - Use some heuristic search strategies

Search Optimization

- Forwarding-based optimization
  - Select subset of neighbors instead of flooding to all neighbors
- Cache-based optimization
  - Index-based cache and content-based cache
  - Replicate index/content across the system

Iterative Deepening

- Multiple breath-first searches (BFS) initiated with successively larger depth limits
  - System policy of multi-depth criteria {a, b, c }; a<b<c
  - Source (S): issue query with depth a
    - query satisfied ?
      - if not, issue resending message of depth b
      - …
    - Stop anyway if hit largest depth limit

K-walker

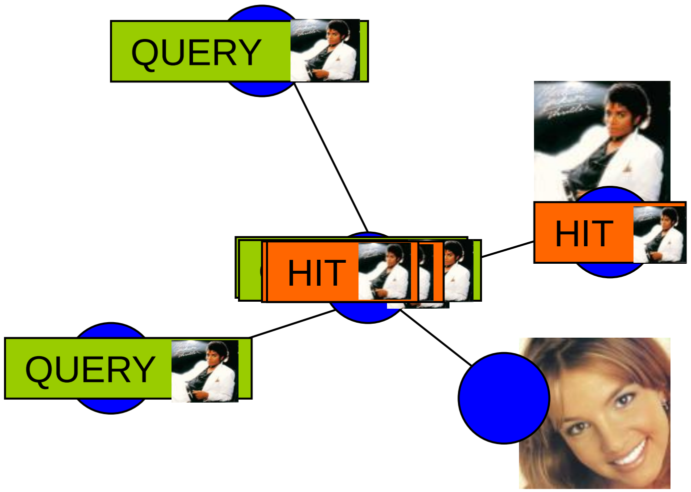
- Each peer forwards a query to k randomly chosen neighbors per step
  - A tradeoff between BFS and DFS
    - Search traffic less than BFS
    - Response time larger than BFS
  - Check back with source before walking to next node
  - Still a random search
  - Can combine with heuristic strategies

Caching and Replication

- Nodes cache copies (or pointers to) content
  - Object info can be pushed from nodes that have copies
  - More copies leads to shorter searches
- Caches have limited size: can't hold everything
- Objects has different popularities: different content requested at different rates

Query Caching

- Technique
  - Nodes may chose to respond to a QUERY message with someone else's QUERYHIT message that was seen in the past.
- Advantages
  - Reduces QUERY traffic for popular searches
- Disadvantages
  - May limit search scope



Query Caching

- Query locality
  - 30% to 40% of queries are repeatedly queries that have been submitted before
- Cached results
  - Valid up to a timeout period
- In Gnutella, a peer can result in 3.7 times reduction in traffic while using a few MBs of storage

File Replication

- Explicit
  - Replicated by system
    - which files are replicated, how many copies, and on which peers to put the copies.
  - Weaken the autonomy of peers
    - A tradeoff between peer autonomy and search efficiency
- Implicit
  - Replication performed automatically by peers requesting and downloading files
    - Random and complicated

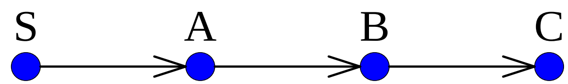
Replication Policy

- How many copies to replicate?
  - Optimal replication policy: square root replication
  - The number of copies is proportional to the square root of the requested rate
  - Cache a copy of object i (once found) at each node visited while searching for object i
    - This will achieve square-root allocation
- Cache replacement
  - Each copy in the cache disappears from the cache at some rate independent of the object content

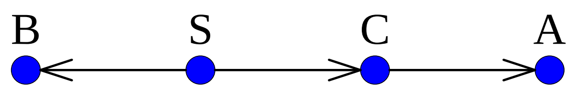
Course Outline

- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
  - Improvement of searching strategies
  - Topology mismatch problem
  - Trustworthy computing
- Summary

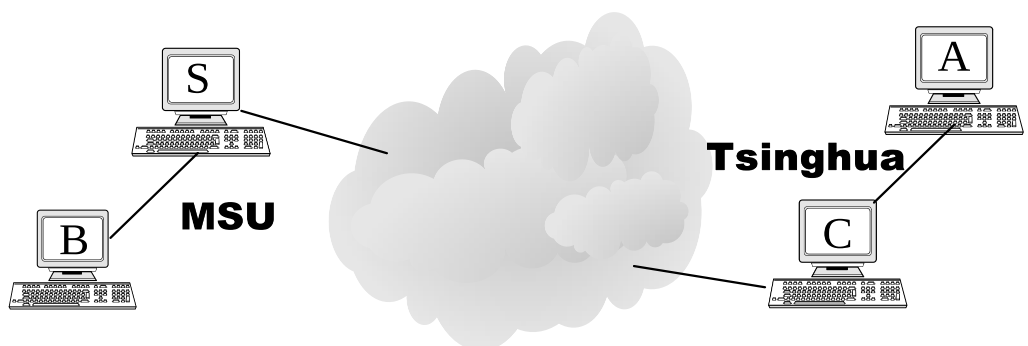
Topology Mismatch Problem



(a) Mismatching Overlay



(b) Matching Overlay



(c) Underlying Physical Topology

S is the source. The longest physical link SC will be traversed three times when the overlay does not match the physical.

What is an Optimal Overlay?

Try to minimize two main Metrics

- Total Traffic Cost: $C = M \times L$

  M is the number of messages that traverse the overlay connection

  L represents the number of physical links in this overlay connection

- Average Query Response Time (Average Distance in an overlay with *p* nodes):

$$AD(G) = \frac{1}{p(p-1)} \sum_{i=1}^{p} \sum_{j=1,\, j \neq i}^{p} v_i v_j$$

Course Outline

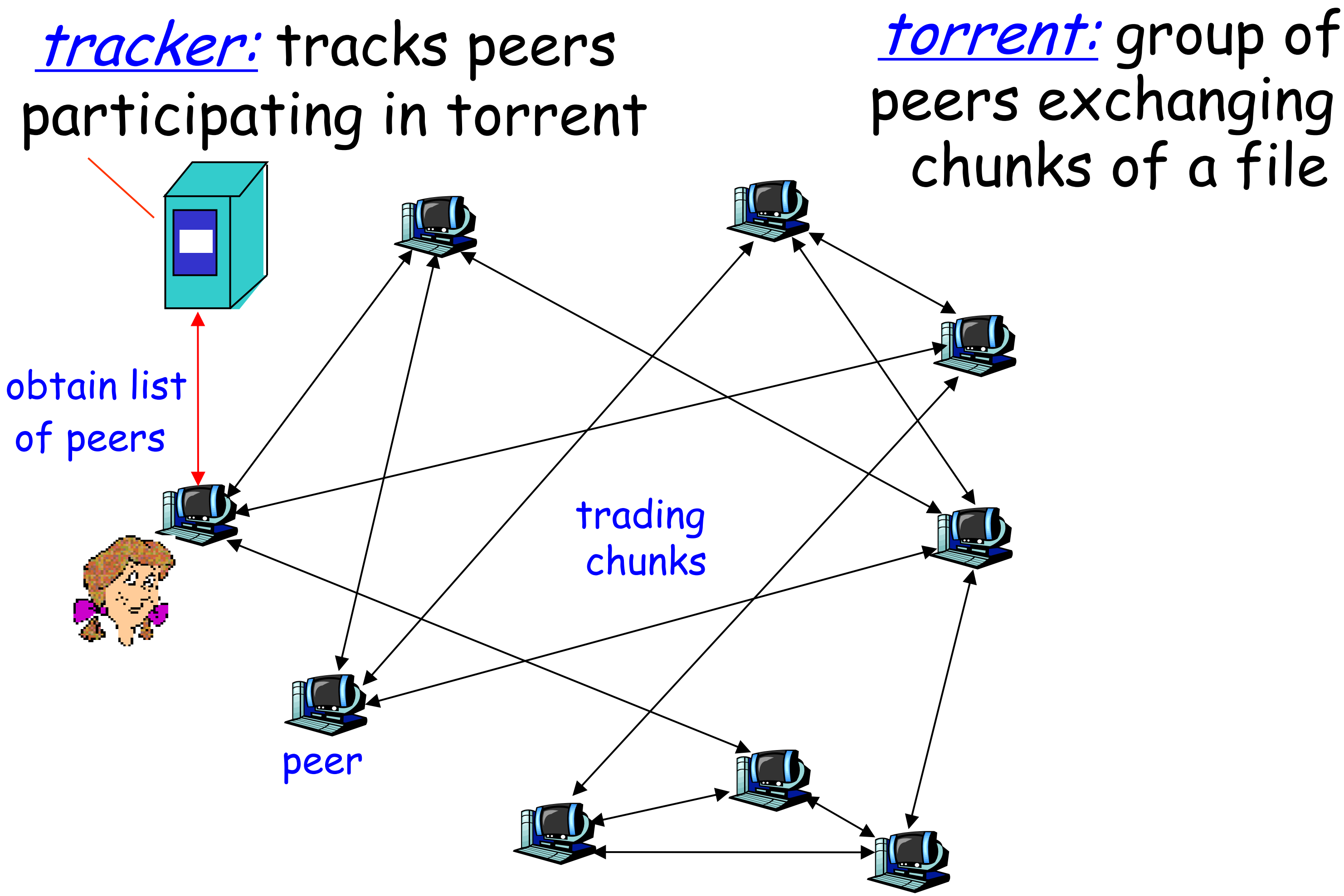
- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
  - Improvement of search strategies
  - Topology mismatch problem
  - Trustworthy computing
- Summary

Free Rider Problem

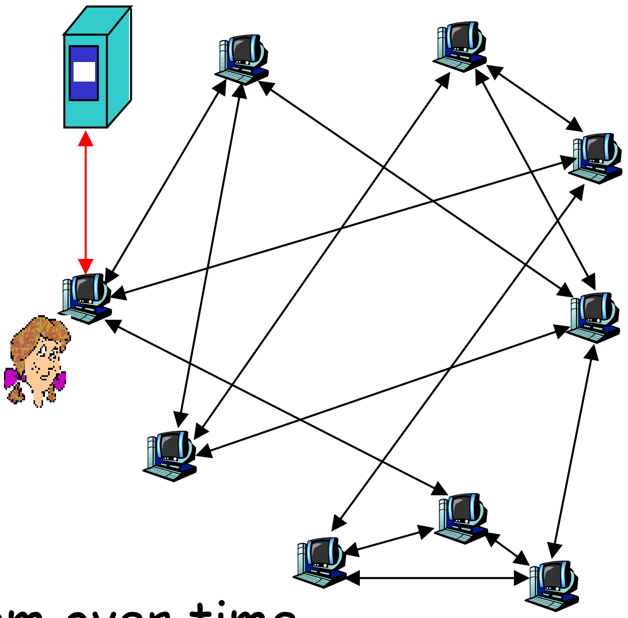
- Free riding
  - No individual is willing to contribute towards the cost of *something* (public goods) when he/she hopes that someone else will bear the cost instead
- In P2P, many people just download files contributed by others and never share any of their files
  - Files shared in P2P are 'public goods'
  - Free Rider Statistics in Gnutella: 70% of users share no files
- Free riding leads to degradation of the performance of the system and adds vulnerability to the system
- Lacks incentives for cooperation
- How to deal with free riders?
  - Case study: BitTorrent

BitTorrent

- P2P file distribution



BitTorrent (1)



- file divided into 256KB *chunks*.
- peer joining torrent:
  - has no chunks, but will accumulate them over time
  - registers with tracker to get list of peers, connects to subset of peers ("neighbors")
- while downloading, peer uploads chunks to other peers.
- peers may come and go
- once peer has entire file, it may (selfishly) leave or (altruistically) remain

BitTorrent (2)

Pulling Chunks

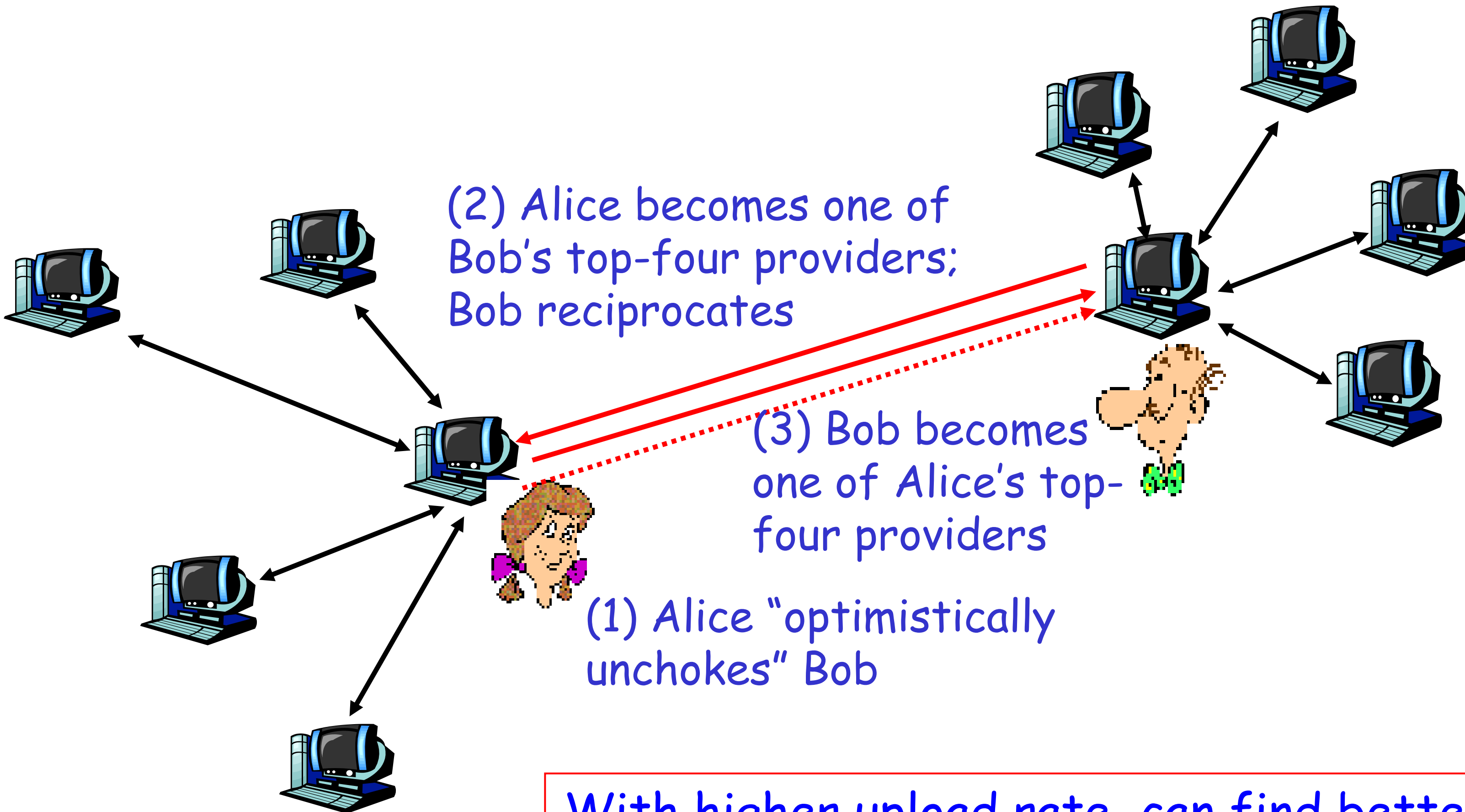
- at any given time, different peers have different subsets of file chunks
- periodically, a peer (Alice) asks each neighbor for list of chunks that they have.
- Alice sends requests for her missing chunks
  - rarest first

What would a free rider do to download chunks?

Sending Chunks: tit-for-tat

- Alice sends chunks to four neighbors currently sending her chunks *at the highest rate*
  - The 4 peers are said to be "unchoked"
  - re-evaluate top 4 every 10 secs
- every 30 secs: randomly select another peer, starts sending chunks
  - newly chosen peer may join top 4
  - "optimistically unchoked"

BitTorrent: Tit-for-tat



With higher upload rate, can find better trading partners & get file faster!
Free riders keep being slow.

Course Outline

- Why P2P?
- A brief history
- P2P architectures
- Usage of P2P technology
- Summary

P2P Computing

- P2P is an interesting paradigm
  - Scalability
  - Easy reconfiguration
  - Potentially simpler application code
- But is it broadly useful?
  - P2P streaming?
- Security issues
  - What if some nodes do not behave?
  - How can attackers hinder operation of some nodes in P2P systems?
  - What can be done to hinder attacks?

P2P Computing (cont'd)

- Are structured P2P systems really useful?
  - Efficient handling of system dynamics
- What other applications can benefit from P2P computing?
  - Tailor applications to P2P model
  - Tailor P2P model to applications
- Need to reduce network traffic
  - Many P2P applications were blocked due to excessive amount of traffic
- Legal issues