Distributed Systems

Peer-to-Peer Systems

Dynamic and Distributed
Information Systems

Today's Agenda

- Historical Overview
- Define P2P
- What are the important issues
- Application Areas
- Brief overview of File Sharing

Historical Overview

Peer-to-Peer Systems

Historical perspective

- 1970s - 1980s: Birth of the Internet
  - Limited reach of the Internet
  - Email, FTP, Telnet
  - Share documents and resources between research centers
  - Central committee to organize and maintain it

- 1990s
  - Tremendous expansion & diffusion
  - Killer apps: WWW and e-Commerce
  - Client/Server model

- Late 1990s - today
  - P2P: An alternative to Client/Server
  - Passive clients → **active peers**
  - End-computers play a role, contribute, interact

How it all started

- June 1999
  - Napster is born / 1st generation of P2P
  - Users not only download content but also provide content to others
  - Users establish a virtual network, entirely independent of physical network and administrative authorities or restrictions
  - Basis: UDP and TCP connections between the peers
- December 1999: RIAA files a lawsuit against Napster Inc.
  - TARGET: the central lookup server of Napster
  - ACHIEVEMENT: Napster popularity skyrocketed!
- February 2001: Peak operation
  - 26.4M users
  - 2.79 billion files / month
- July 2001: Judge orders Napster to pull the plug!
  - Napster network breaks down instantly

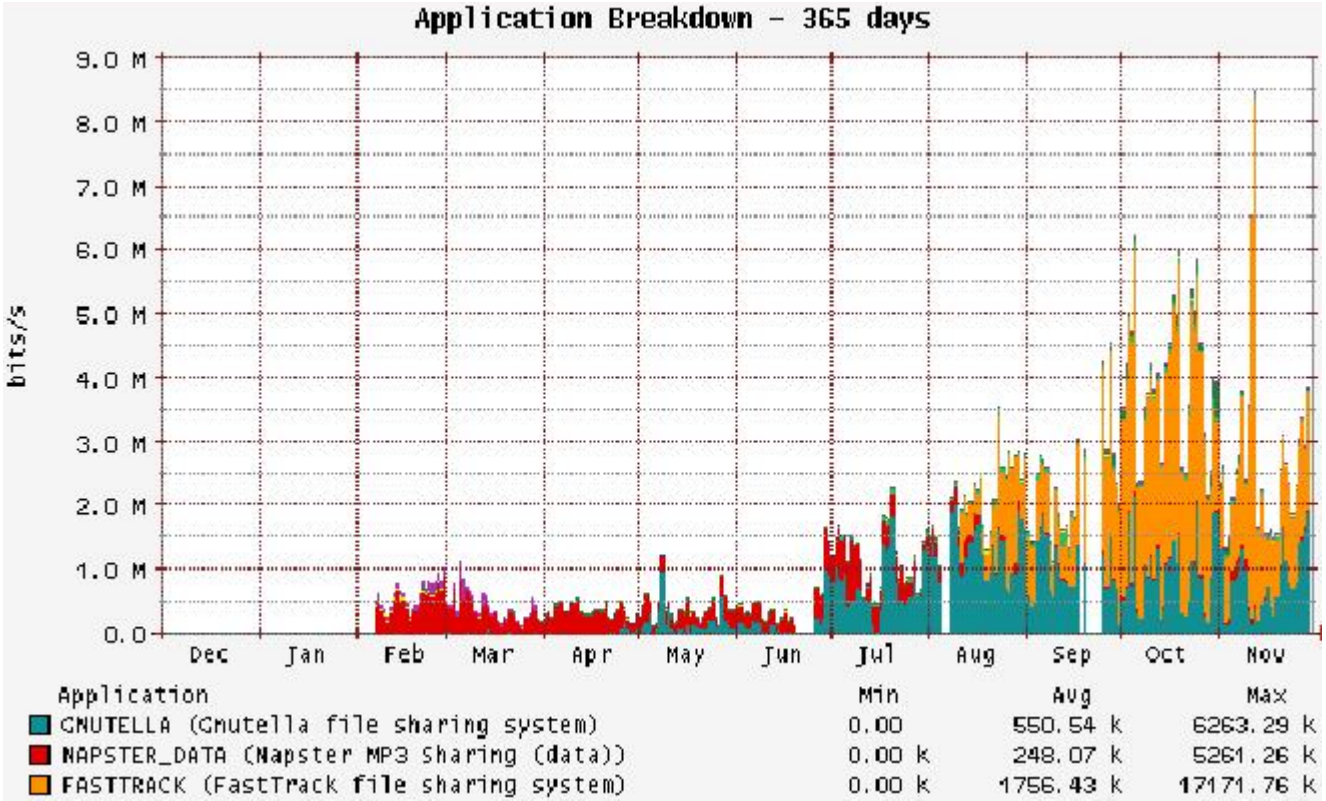
How it continued

- March 2000
  - Nullsoft releases Gnutella as an open source project
  - Fully decentralized
  - Additionally to offering files, the peers also take over routing tasks
  - No central lookup server → no single point to attack

- Later in 2000: Superpeer concept
  - Hierarchical routing layer
  - Significantly improves scalability and efficiency
  - FastTrack (Morpheus, KaZaA)
  - eDonkey2000

- 2001 - 2002
  - KaZaA loses ground (many defected files due to weak hash keys to identify files)
  - eDonkey and Gnutella regain popularity
  - eDonkey becomes most popular file-sharing network: 2-3M *online* users
  - Gnutella v0.6 adopts superpeer architecture (*ultrapeers* in Gnutella terminology)

P2P Traffic in 2001



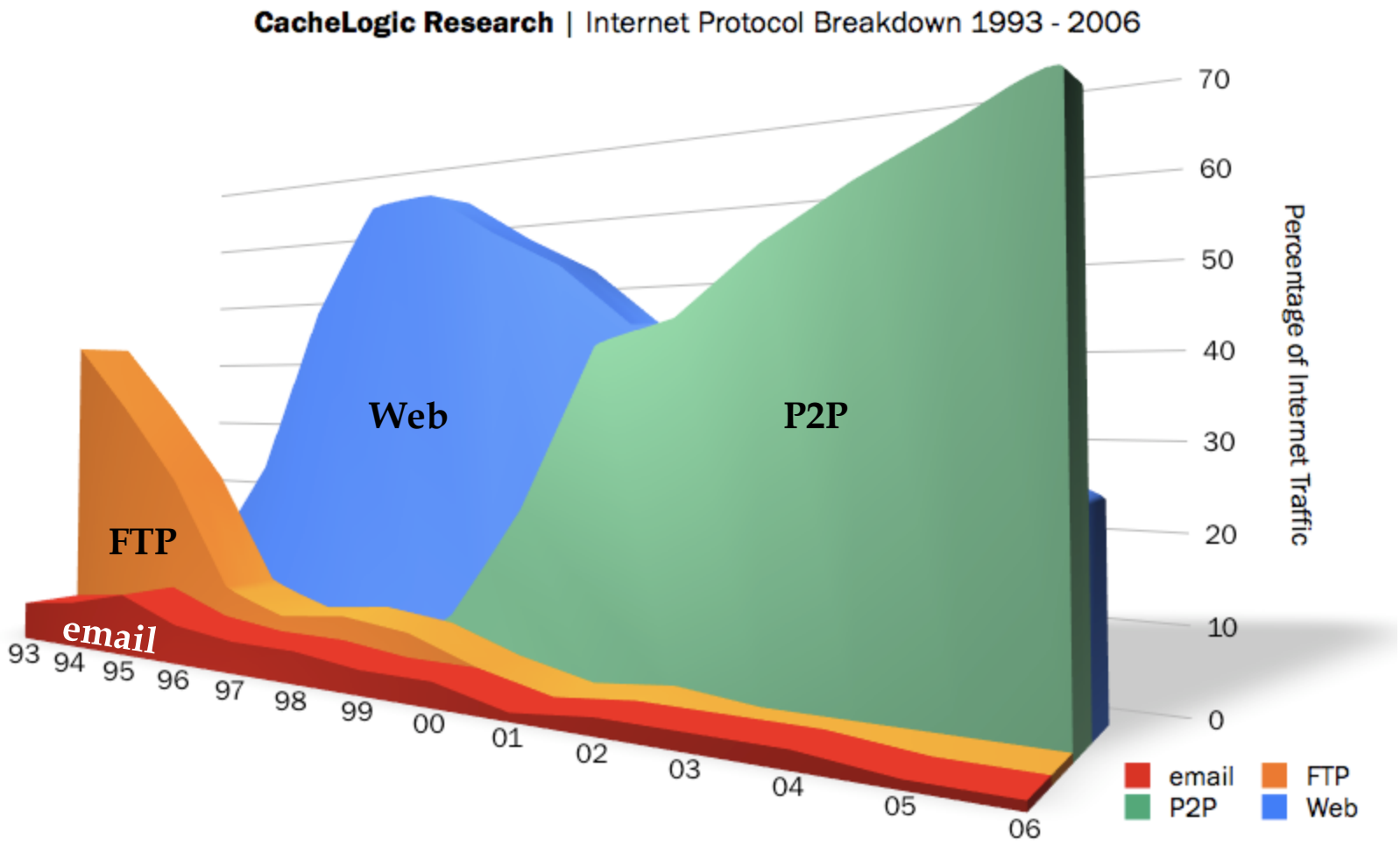
How it took off!

- 2002
  - First version of BitTorrent released

- 2003
  - BitTorrent causes majority of the observed traffic
  - Downloads significantly faster, due to mechanism against free-riding
- Middle of 2003
  - New P2P concepts develop
  - Skype is born: a P2P Voice-over-IP application

- In the meantime: More P2P domains explored!
  - P2P Routing
  - Network Storage
  - P2P Multicasting
  - Data aggregation
  - P2P Streaming
  - etc.
- Today:
  - Major efforts are made to increase the reliability of P2P systems, to use P2P also in mobile networks, etc.

Internet Traffic



Define P2P

Dynamic and Distributed
Information Systems

A simple definition

"Endpoints talk directly to each other, as opposed to client/server"

INACCURATE


E-mail


IP Routing


Telephones!


NAPSTER: Based on a centralized server!!!

What makes P2P interesting?

- End-nodes are promoted to active components!
  - previously they were just clients
- Nodes participate, interact, contribute to the services they use.
- Harness huge pools of resources accumulated in millions of end-nodes.

Is application *XYZ* P2P?

- Do nodes contribute to the system?
- Do nodes collectively carry out a service?
- Are variable connectivity and temporary network addresses the norm?
- Do nodes have significant autonomy?
- Can they (generally) be heterogeneous?
- Who owns the hardware?
  - Single-administered entity?
  - Distributed among participating users?

A better definition

P2P is a class of systems where:

- Resources available at the edges of the Internet are utilized:
  - Storage
  - CPU cycles
  - Bandwidth
  - Content
  - Human presence
- Service is carried out collectively
  - Nodes share both benefits **and** duties
- Irregularities and dynamicity are treated as the norm

Dual nature: Client & Server

client

server

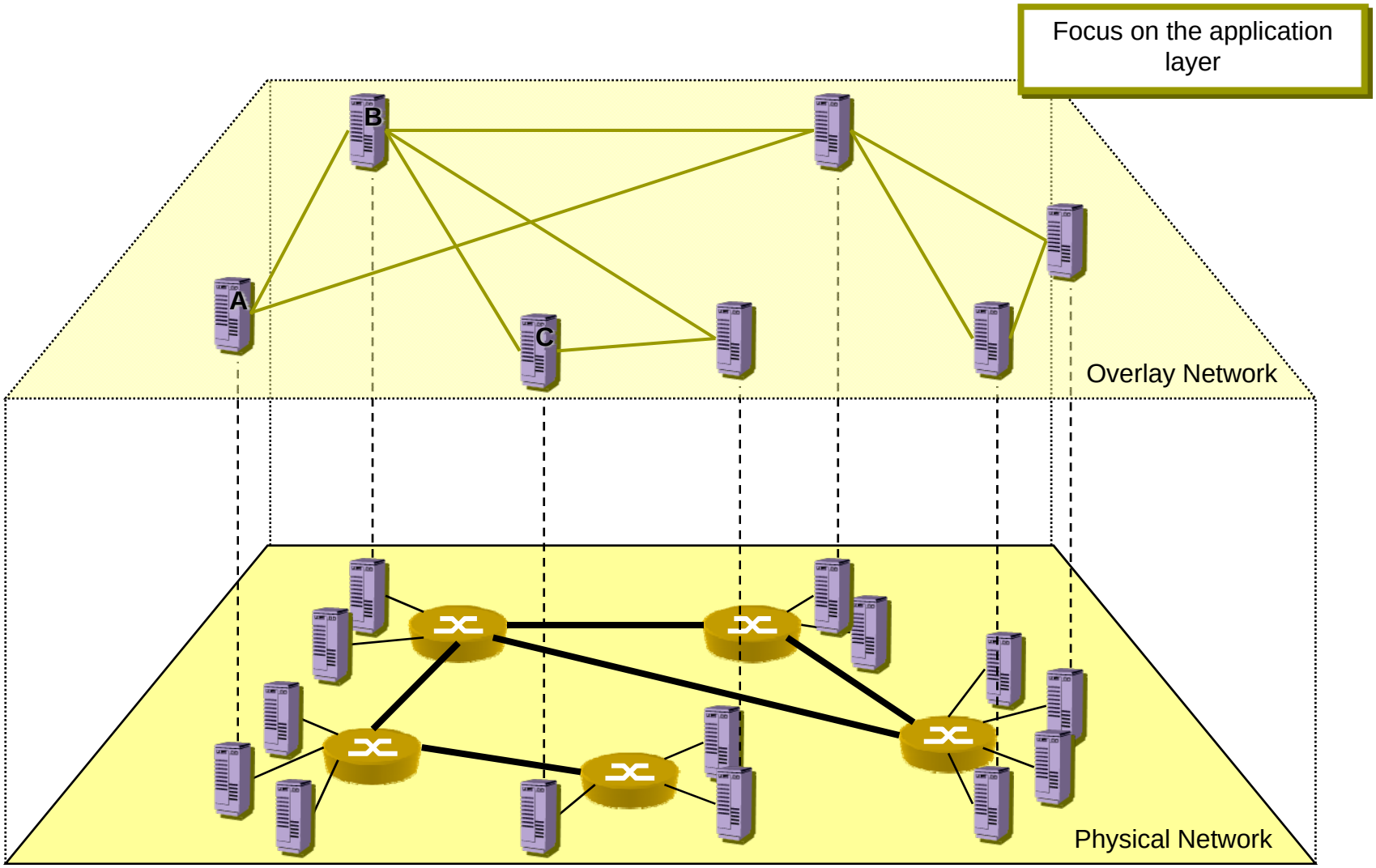
Main advantages of P2P

- Inherently scalable:
  - higher demand → higher contribution!
- Increased (massive) aggregate capacity
- Utilize otherwise wasted resources
- Distribute load and administration
- Designed to be fault tolerant
- Inherently handle dynamic conditions

Important Issues in P2P

Dynamic and Distributed
Information Systems

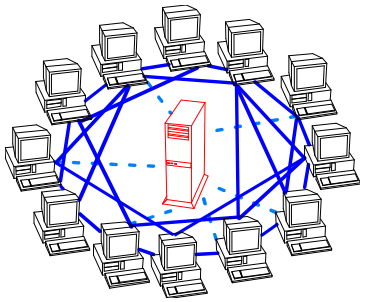
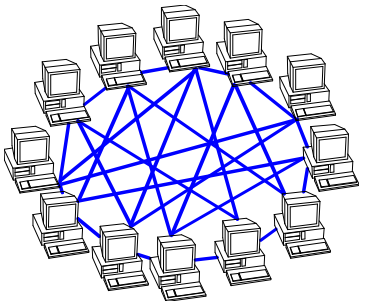
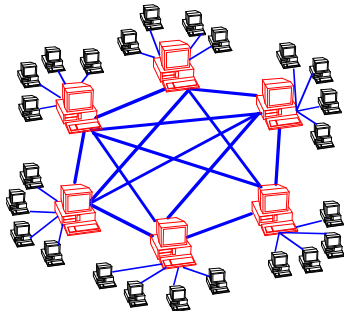
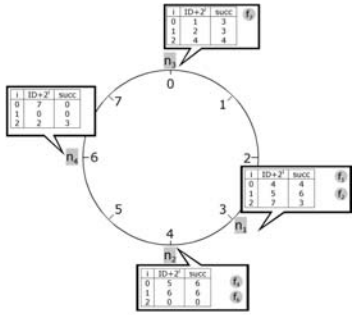
Overlay Networks



Overlay types

Unstructured P2P	Structured P2P
■ Any two nodes can establish a link ■ Topology evolves at random ■ Topology reflects desired properties of linked nodes	■ Topology strictly determined by node IDs

Overlay types

Unstructured P2P			Structured P2P
Centralized P2P	**Pure P2P**	**Hybrid P2P**	**DHT-Based**
Central entity necessary to manage the overlay	No central entities	Multiple & Dynamic central entities	No central entities
Central entity is some kind of index/group database	Any node can be removed without loss of functionality	Any node can be removed without loss of functionality	Fixed links, determined by node IDs
Example: Napster	Example: Gnutella v0.4, Freenet	Example: Gnutella v0.6, Freenet	Any node can be removed without loss of functionality
			Examples: Chord, Pastry, CAN
			

Main Issues in P2P

Overlay Maintenance

- Bootstrapping
  - how to join the system
- Continuous maintenance
  - how to handle changes, faults, etc.

Main Issues in P2P

Scalability

- Avoid central server!
- Distribute load on multiple peers
- Limit load per peer
  - Computing
  - Messaging
  - Storage
  - State

Main Issues in P2P

Fairness

- Load balancing
- Distribute load among peers, but how?
  - Evenly?
  - Proportionally to node capacity?
  - …?
- User behavior!
  - users are selfish and independent (maximize own benefit)
  - give incentives for fair play
  - to maximize benefit → abide by the rules!

Main Issues in P2P

Dynamicity and Adaptability

- Changing topology
  - nodes join and leave: ***node churn***
  - network partitions
- Changing data
  - content is changed
  - files are added / deleted
- Changing profiles
  - users change interests
  - new semantic categories introduced
- Change in load
  - load rebalancing

Main Issues in P2P

Fault Tolerance

- Robustness of the overlay
- Resilience to failures
- Resistance to node & link crashes
- Availability

Main Issues in P2P

Self-Organization

- Key for
  - Overlay maintenance
  - Adaptability
  - Fault Tolerance
  - Robustness
- No one keeps full state: nodes take local decisions
- Globally smooth operation should emerge from local decisions!!
  - Self-Management
  - Self-Healing
    - repair problems encountered
  - Self-Configuration
  - Self-*

Main Issues in P2P

Performance

- Efficiency
  - in searching
  - in routing steps
  - in discovering relationships
  - etc.
- Locality
  - reduce network latency

Main Issues in P2P

Privacy

- Anonymity
  - …who downloaded a copyrighted movie?
  - …who wrote the bad review about Spyros' course?
- Reputation
- Resistance to censorship

Main Issues in P2P

Security

- Defend against DDOS attacks
- Disseminate worm protection patches: Speed is crucial!
- Make P2P systems themselves secure

Main Issues in P2P

Legal issues

- Copyright violation
- Direct infringement
  - e.g., download or upload copyrighted files
- Indirect infringement
  - e.g., someone offers the means for direct infringement

Main Issues in P2P

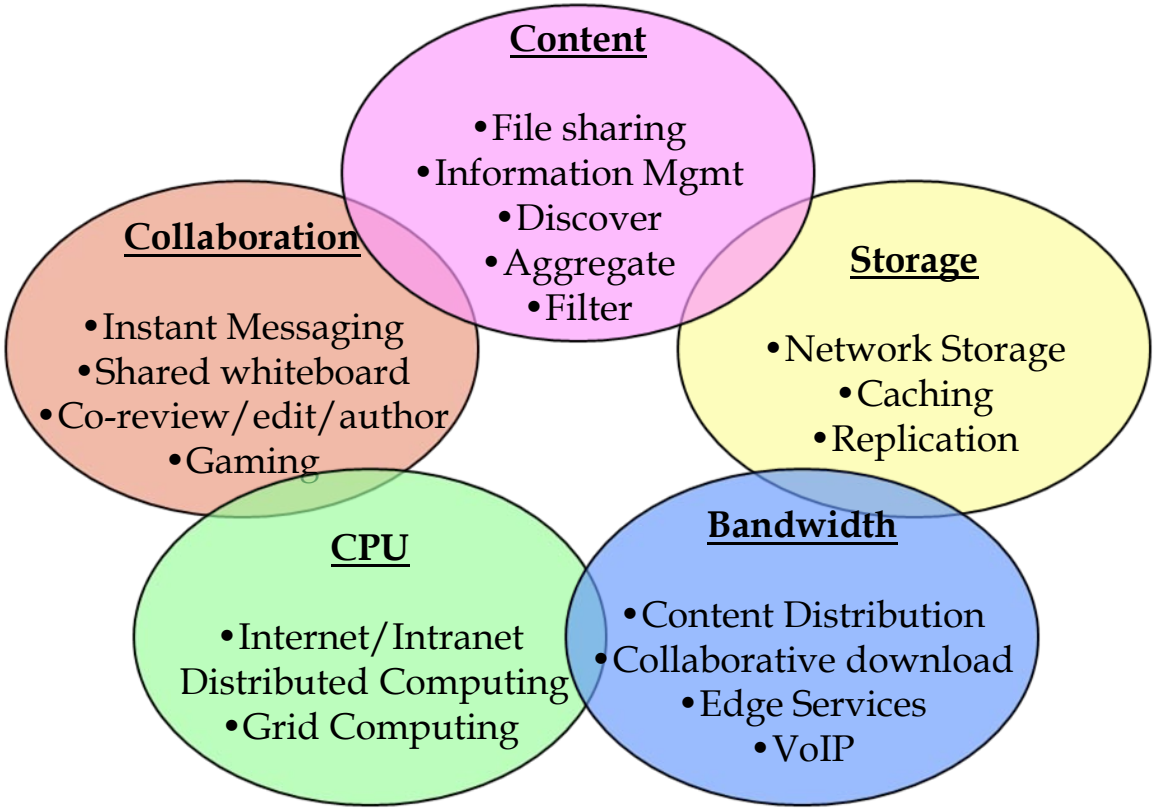
SIMPLICITY !

- Things can easily get out of control with thousands of nodes under dynamic conditions!

Application Areas

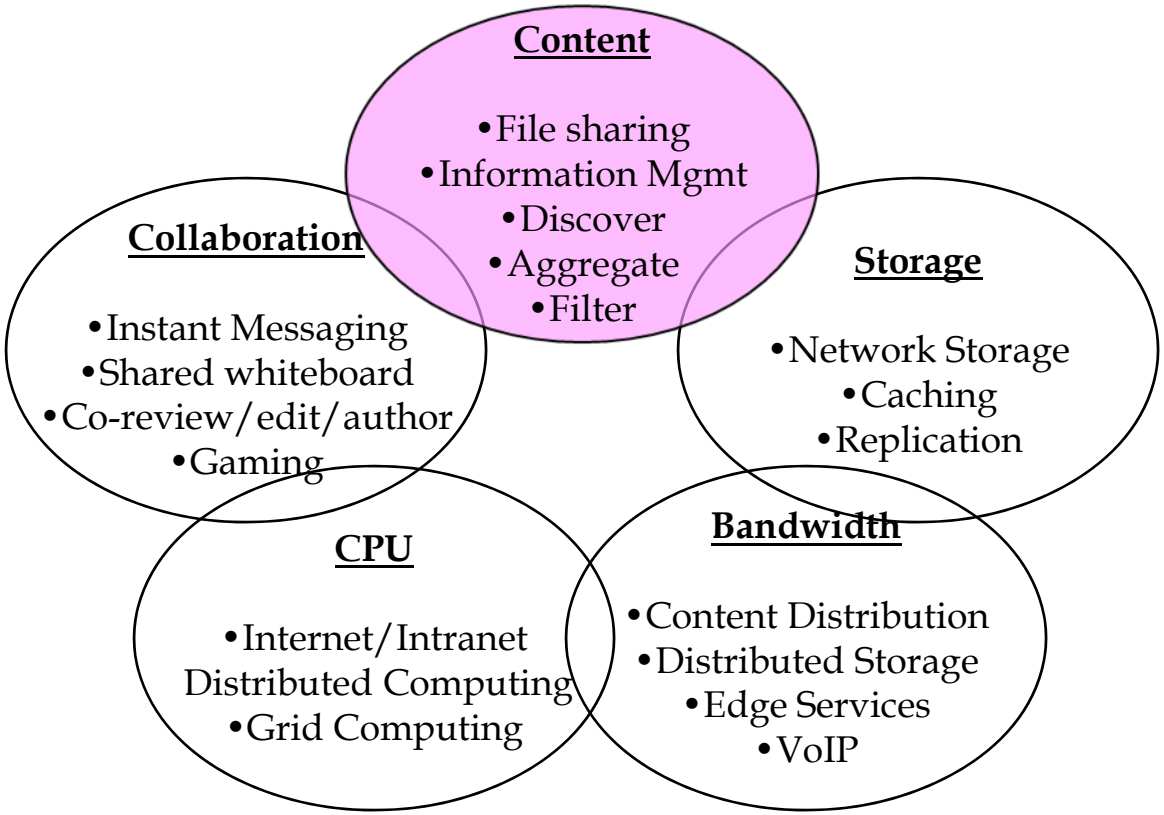
P2P Application Areas

High-level grouping of P2P apps based on shared resource

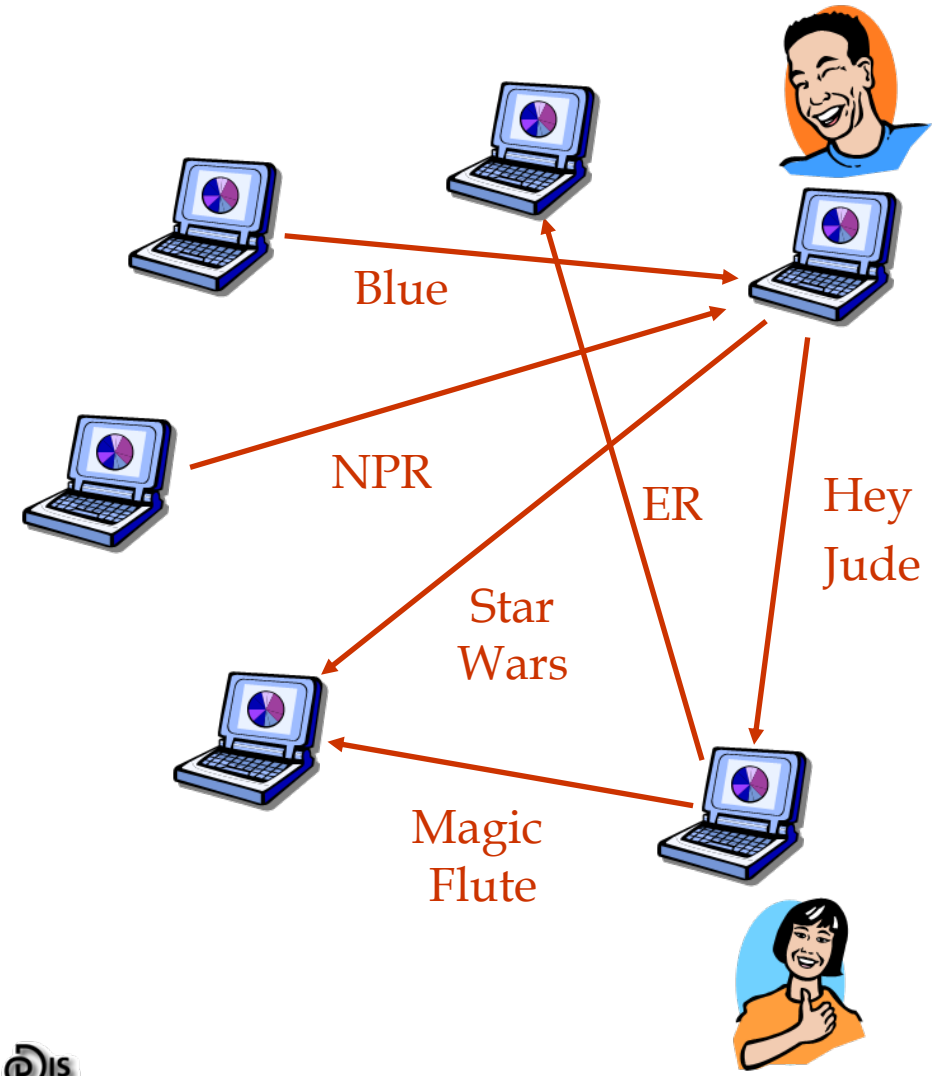


P2P Application Areas

High-level grouping of P2P apps based on shared resource



Sharing Content



Killer deployments

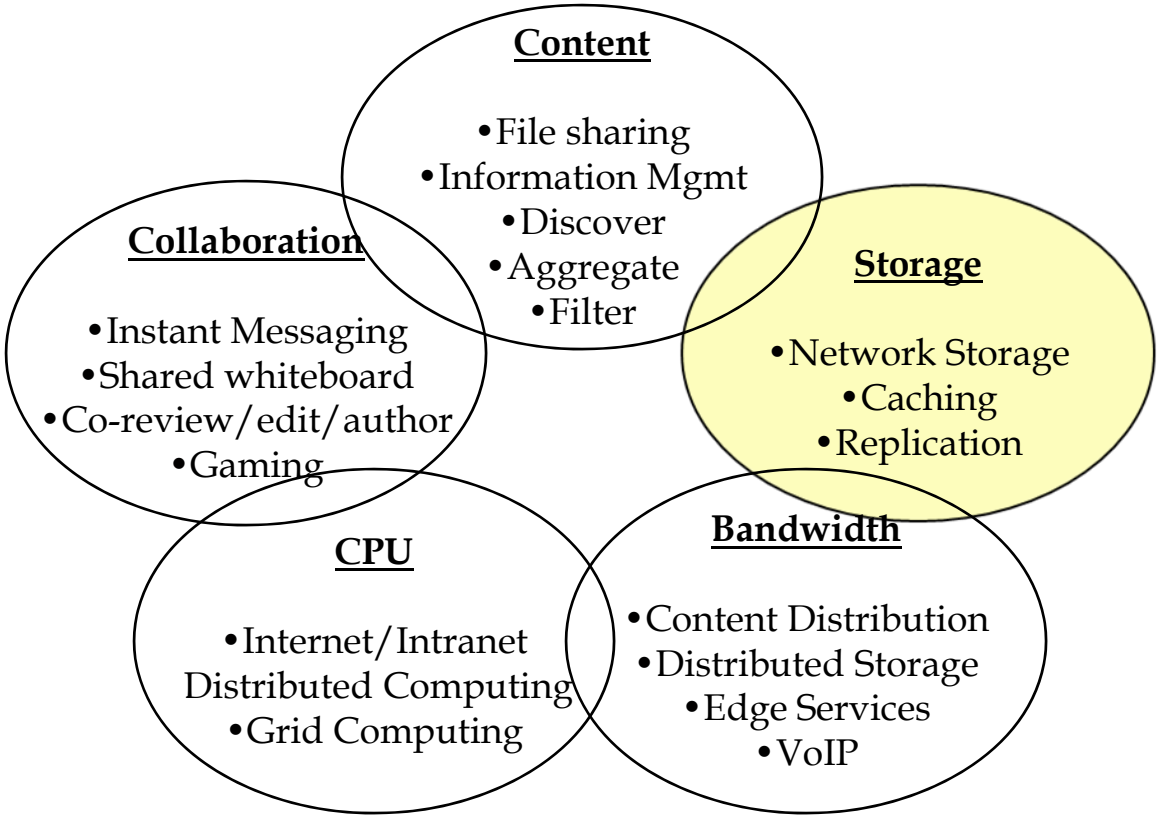
- **Napster**
- **Gnutella**
- **KaZaA/FastTrack**
- **eDonkey2000**
- **BitTorrent**

- Large distributed storage
- Very high variation of content

- Unstable availability
  - No guarantees

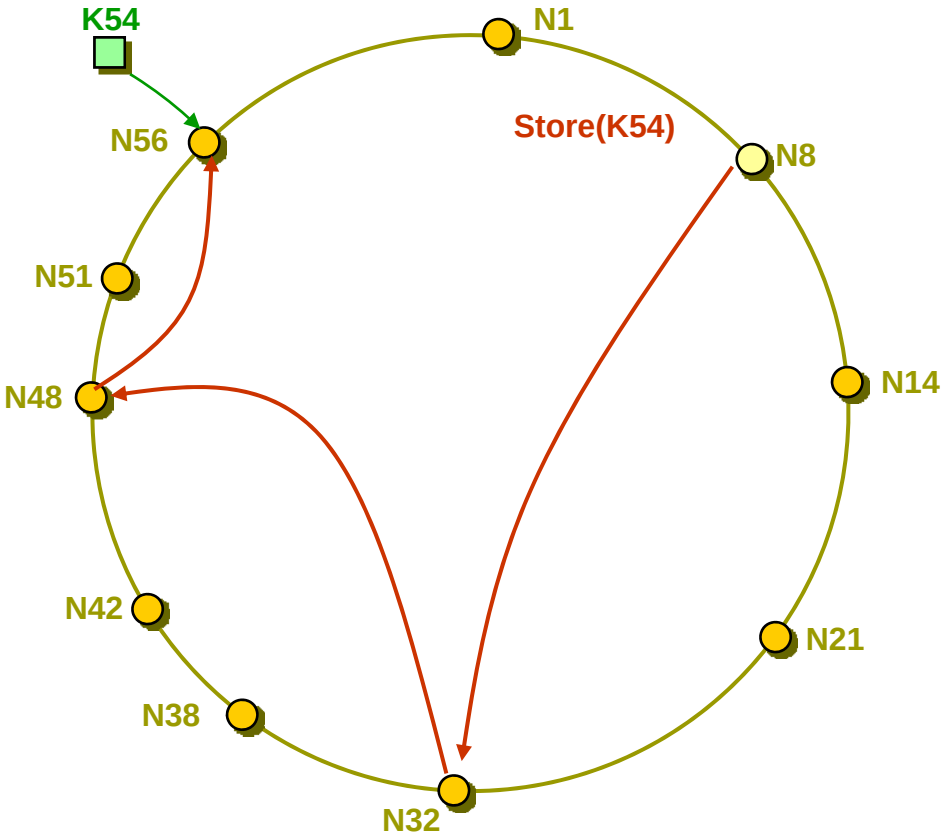
P2P Application Areas

High-level grouping of P2P apps based on shared resource



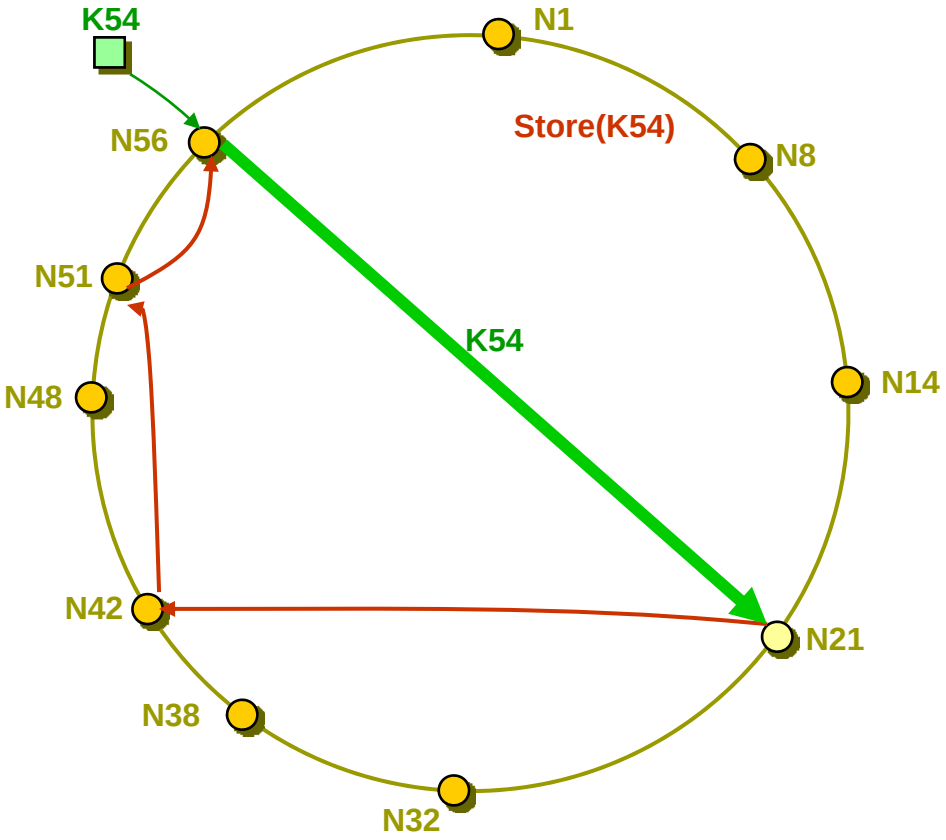
Network Storage

- OceanStore
- PAST



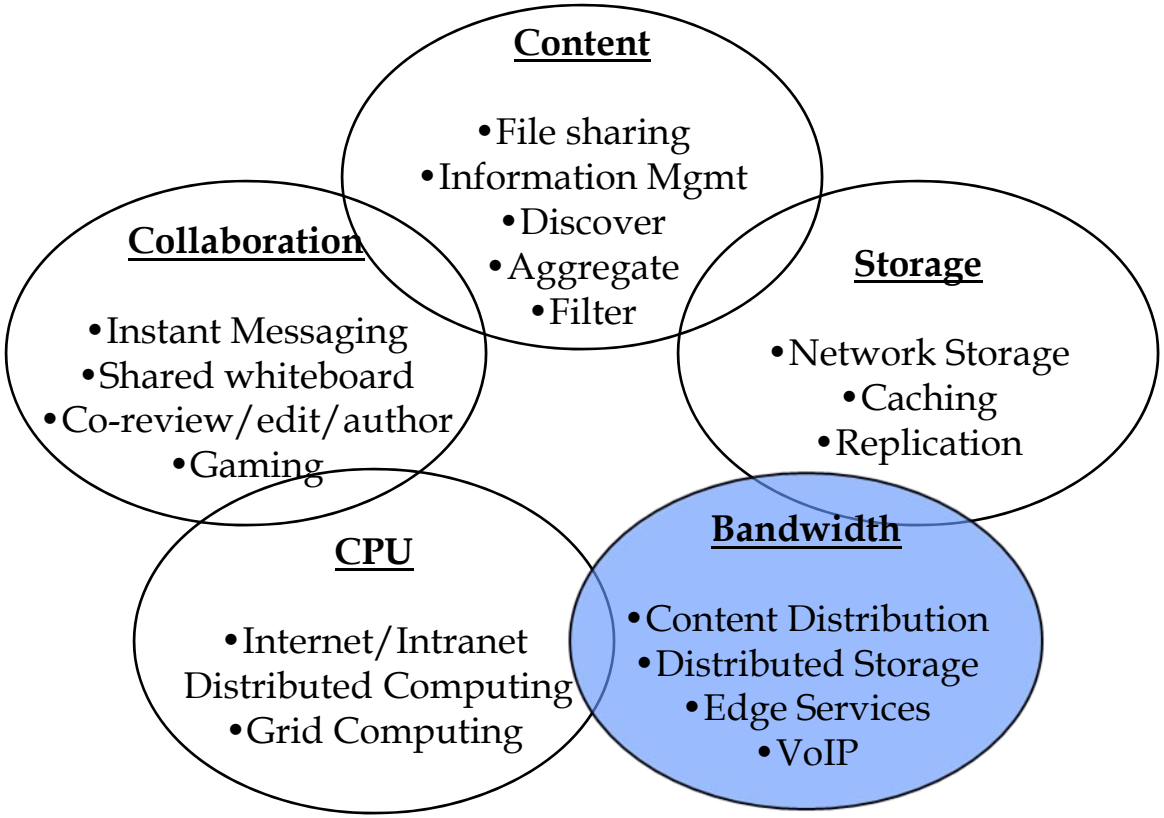
Network Storage

- OceanStore
- PAST



P2P Application Areas

High-level grouping of P2P apps based on shared resource



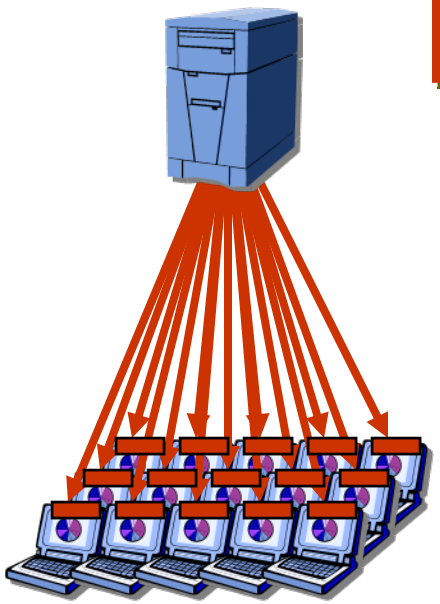
Contributing Bandwidth

- CDNs (Content Distribution Networks)
- BitTorrent
- File-sharing systems

Contributing Bandwidth

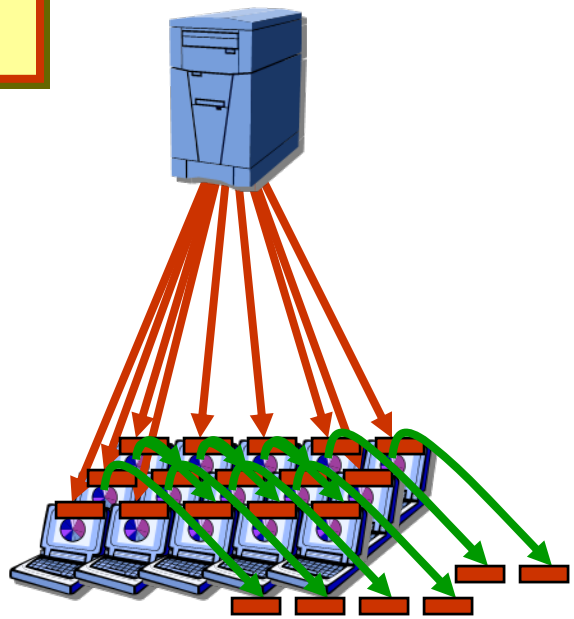
Source server: **100 Mb/s**
Clients: **10 Mb/s**
1. Antivirus update
100,000 clients
File: **4 MB**
2. Daily database update
1000 clients
File: **600 MB**

1. 9h:52m
2. 14h:48m



Client/Server

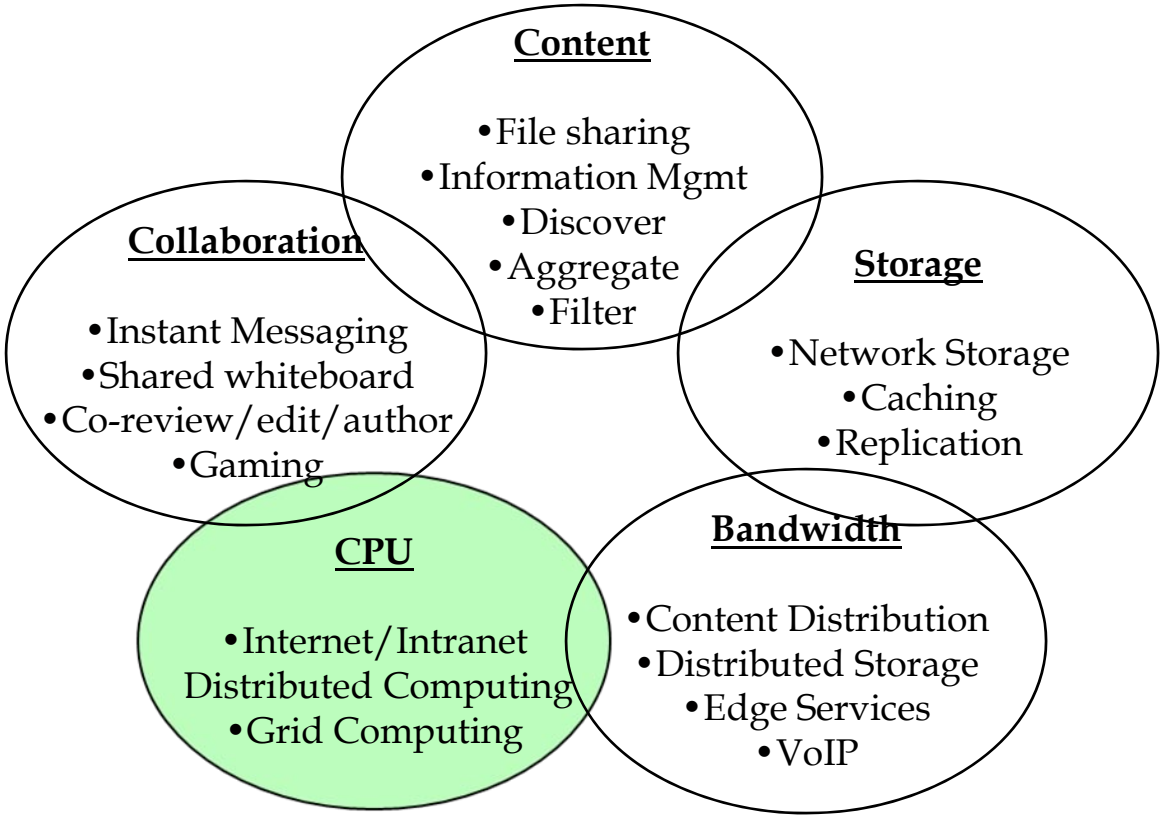
1. 52s
2. 09m:54s



Cooperative

P2P Application Areas

High-level grouping of P2P apps based on shared resource



Sharing CPU

- Increasing requirements for High Performance Computing
  - i.e., in the field of bio-informatics, logistics or the financial sector
- Available computing power of endpoints often unused
- Use P2P to bundle processor cycles:
  - Forming a cluster of independent, networked computers that are combined into a single logical computer
  - Achieve computing power which even the most expensive super-computers can scarcely provide
  - *"Grid Computing"*

Sharing CPU --- Examples

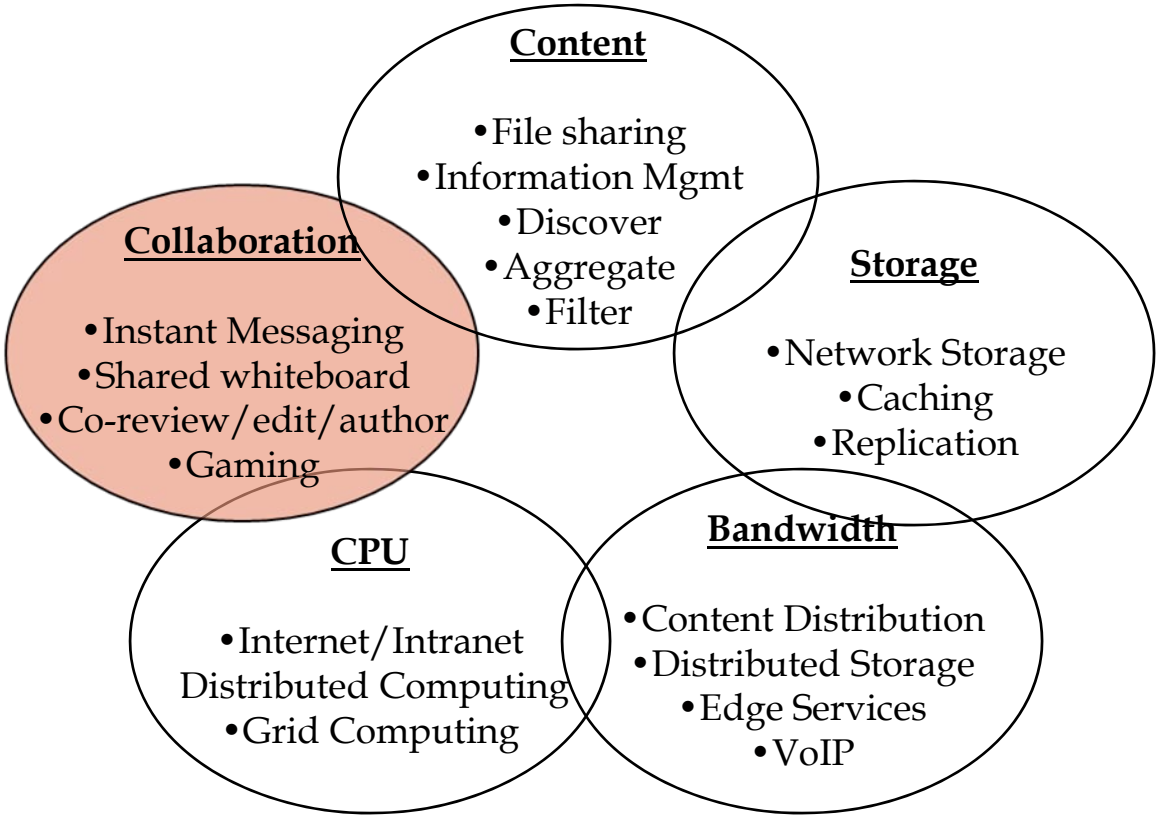
- Popular example: *SETI@home*
  - Calculations during the idle processor cycles of participating peers.
- Successors:
  - BOINC (Berkeley): http://boinc.berkeley.edu
  - World Community Grid (IBM) : http://www.worldcommunitygrid.org
    - Biology and Medicine
    - Climate simulations
    - Math
    - Astronomy, Physics, Chemistry

NOTE:
The core of these systems is a classical Client/Server application

- Advanced vision of grid computing: *Globus Toolkit*
  - Standardized middleware for grid application.

P2P Application Areas

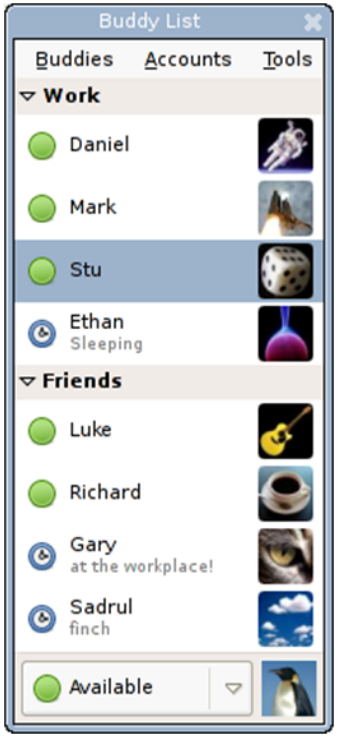
High-level grouping of P2P apps based on shared resource



Presence Information

- Presence Information
  - information about which peers and which resources are available

- Example: *Instant Messaging Systems*
  - P2P application which essentially uses presence information
  - Peers pass on information via the network, whether or not they are available for communication



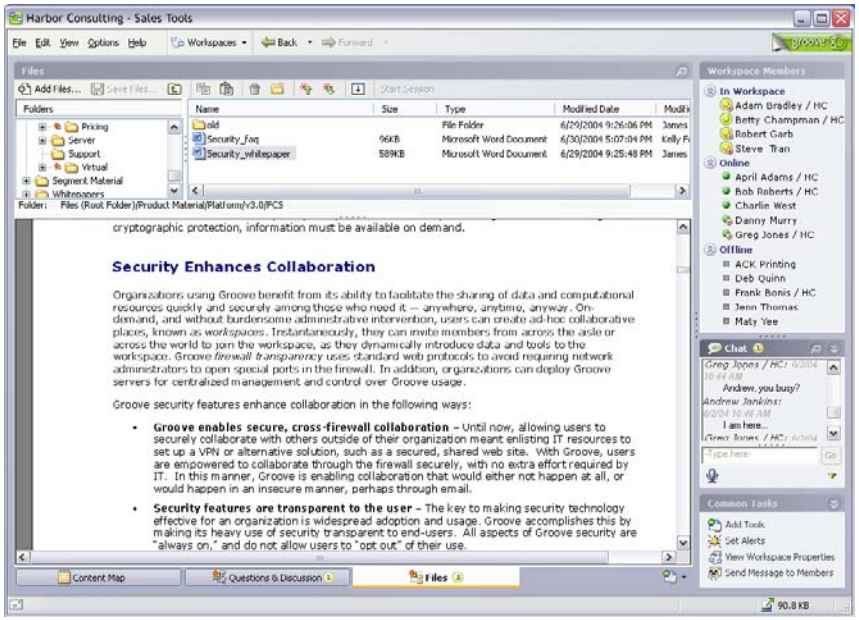
Document Collaboration

- Usually centrally organized
- But
  - In many cases, documents distributed across desktop PCs
  - no central repository having any knowledge of their existence
- Solution
  - P2P networks which create a connected repository from the local data on the individual peers.
  - Indexing and categorization of data by each peer on the basis of individually selected criteria.
  - Self organized aggregation of information from areas of knowledge.

http://www.nextpage.com/

Collaboration

- Collaboration
  - synchronous communication
  - online meetings
  - edit shared documents.
- Groupware
  - offers functions like IM, file sharing, notification, co-browsing, whiteboards, voice conferences and databases with real time synchronization.
  - Client/Server groupware has to be set up and administered for each working group
- P2P Groupware
  - avoid additional administrative task and central data management:
  - All of the data created is stored on each peer and is synchronized automatically.
  - Users can set up shared working environment for virtual teams (so-called shared spaces).
  - Users can invite other users to work in these teams.



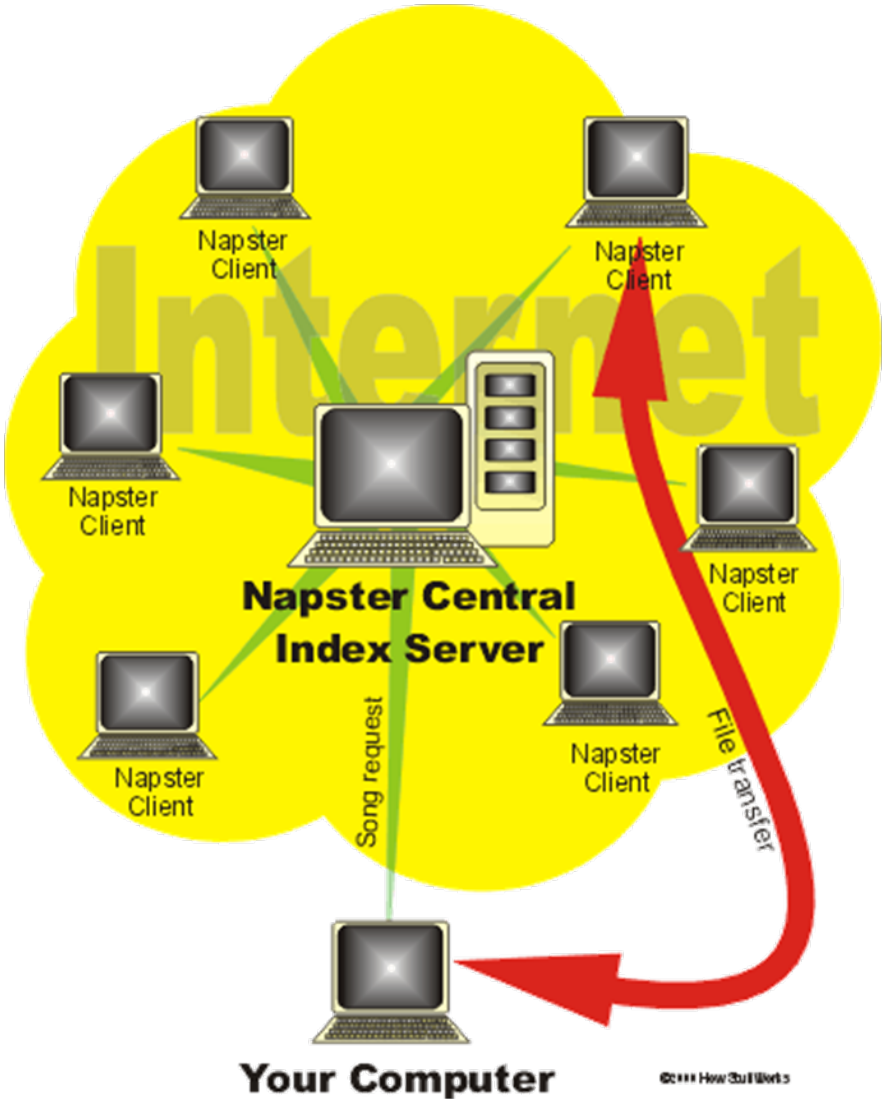
http://www.groove.net/

Basics in File-Sharing

Dynamic and Distributed
Information Systems

Napster: Centralized P2P

- Peer-to-peer
  - relies on a central index
  - but files don't reside on a central server
- Four steps:
  - Connect to Napster server
  - Upload your list of files (push) to server
  - Give server keywords to search the full list
  - Select "best" of correct answers (based on pings)

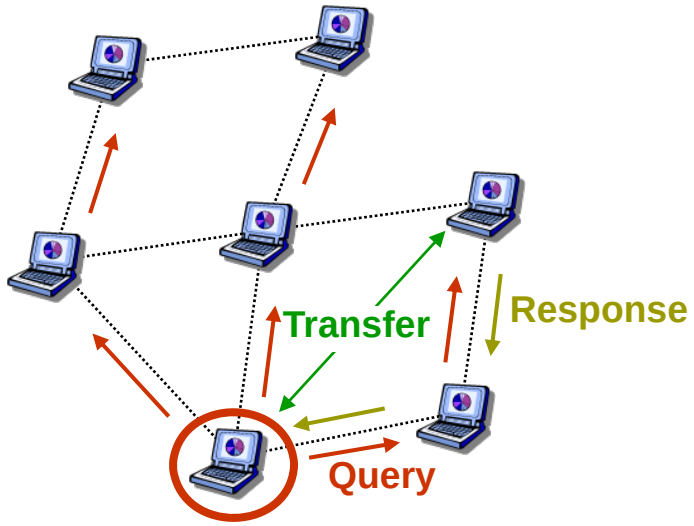


Napster: Clever Design

- Centralized user and song database
  - Quick searching
    - Faster/better than Gnutella
  - Users come and go
    - User/search database continually updated
  - Automatic file sharing
    - Easy to use file server
- ***But…***
  - Single server to bring down
  - This centralization is ultimately its downfall

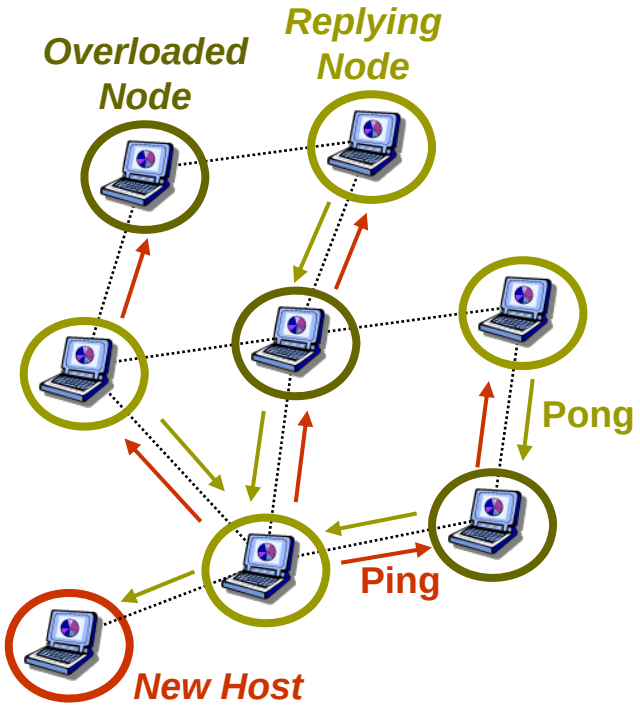
Gnutella: Pure P2P

- Focus: decentralized method of searching
  - harder to "pull the plug"
- Search by flooding
  - If you don't have the file you want, query 7 of your partners (neighbors)
  - If they don't have it, they contact 7 of their neighbors, for a maximum hop count of 10
  - Requests are flooded — may lead to scalability problems
  - No looping but packets may be received twice
- Querying node is sent responses with list of matching files and IP addresses
- File transfer is direct (no anonymity)



Gnutella: Overlay Maintenance

- Plug-in to a host and send a *broadcast ping*
  - Can be any host (hosts transmitted through word-of-mouth or host-caches)
  - Host broadcasts ping message with TTL of 7
- Hosts that are not overloaded respond with a *routed pong*
  - Gnutella caches IP addresses of replying nodes



Gnutella: Problems

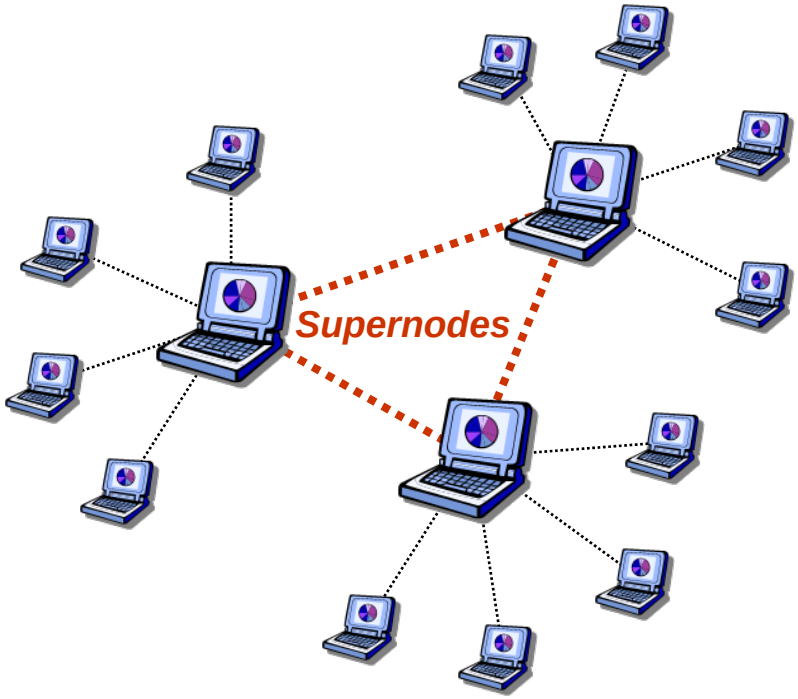
- 24 hour survey showed:
  - 70% of people shared no files
  - 50% of search responses from top 1% of hosts
  - Reverting to client/server
    - Suddenly not so hard to shut down!
  - Verified hypotheses
    - H1: A significant portion of Gnutella peers are free riders
    - H2: Free riders are distributed evenly across domains
    - H3: Often hosts share files nobody is interested in
- Non-standard implementation
  - People implement their own Gnutella clients
  - Some clients are dodgier than others

KaZaA: Hybrid P2P

- Software
  - Proprietary
  - Files and control data encrypted
  - Everything in HTTP request and response messages
- Architecture
  - Hierarchical
  - Cross between Napster and Gnutella

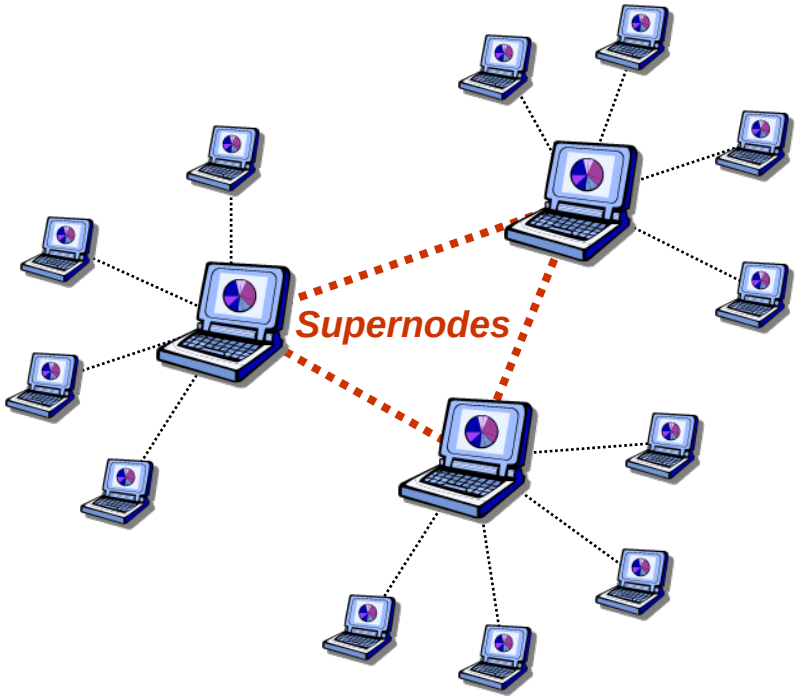
KaZaA: Architecture

- Each peer is either a supernode or is assigned to a supernode
  - Nodes with more bandwidth and that are more available are designated as supernodes
  - Each supernode knows about many other supernodes (almost mesh overlay)
  - Supernodes act as mini-Napster hubs tracking the content and IP addresses of their descendants
  - Guess: ~10,000 supernodes with 200-500 descendants each
  - Dedicated user authentication server and supernode list server



KaZaA: Queries

- Node first sends query to supernode
  - Supernode responds with matches
  - If *x* matches found, done
- Otherwise, supernode forwards query to subset of supernodes
  - If total of *x* matches found, done
- Otherwise, query further forwarded
  - Probably by original supernode rather than recursively



KaZaA: Overlay Maintenance

- List of potential supernodes included within software download
- New peer goes through list until it finds operational supernode
  - Connects, obtains more up-to-date list
  - Node then pings 5 nodes on list and connects with the one with smallest RTT
- If supernode goes down, node obtains updated list and chooses new supernode

KaZaA: Corporate Structure

- Software developed by FastTrack in Amsterdam
- FastTrack also deploys KaZaA service
- FastTrack licenses software to Music City (Morpheus) & Grokster
- Later, FastTrack terminates license, leaves only KaZaA with killer service
- International "cat-and-mouse" game
- Summer 2001, Sharman networks, founded in Vanuatu (small island in Pacific), acquires FastTrack
  - Board of directors, investors: secret
- Employees spread around, hard to locate
- Code in Estonia