ProGres, Part III: Peer-to-Peer

Fabien Mathieu
(fabien.mathieu@normalesup.org)

Master ProGres
Sorbonne Université(s)

November 25, 2022

Roadmap

1. Part I: Connectivity
2. Part II: Web services
3. Part III: Peer-to-peer
   - A brief overview
   - Distributed Hash Tables

Online slides

`https://www-npa.lip6.fr/~tixeuil/m2r/pmwiki.php?n=Main.PROGRES`

Part III

A) What is P2P?

Definition by enumeration

"Everyone" knows what P2P is:

FileSharing	BitTorrent, eMule,...
Streaming	TVants, PPLive,...
VoIP	Skype

Definition by enumeration

"Everyone" knows what P2P is:

FileSharing	BitTorrent, eMule,...
Streaming	TVants, PPLive,...
VoIP	Skype

"Everyone" knows what P2P is not:

Unplugged	Photoshop, Office...
Plugged	Apache, SSH, browsers...

Definition by enumeration

"Everyone" knows what P2P is:

FileSharing	BitTorrent, eMule,. . .
Streaming	TVants, PPLive,. . .
VoIP	Skype

"Everyone" knows what P2P is not:

Unplugged	Photoshop, Office. . .
Plugged	Apache, SSH, browsers. . .

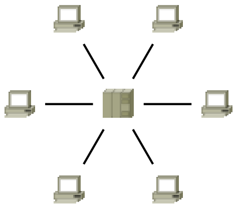
The blurry frontier:

Distributed computing	BOINC projects (Seti@home, Folding@home. . .), BitCcoin. . .
Clouds/datacenters	Amazon EC2/AWS, Microsoft Azure, Dropbox, . . .

Network definition

Most network apps follow a client/server paradigm:

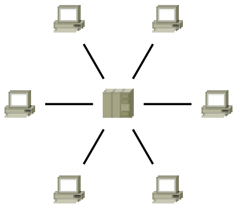
- On the one hand, servers supply resources,
- On the other hands, clients uses these resources.

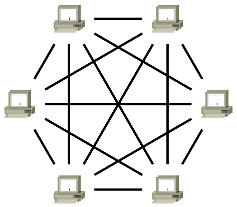


Network definition

Most network apps follow a client/server paradigm:

- On the one hand, servers supply resources,
- On the other hands, clients uses these resources.



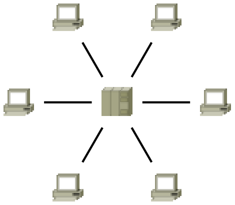


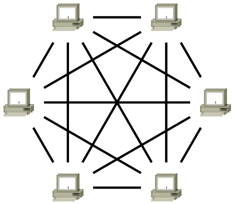
Network definition → break the distinction between client and server: every peer can act as a client and/or as a server.

Network definition

Most network apps follow a client/server paradigm:

- On the one hand, servers supply resources,
- On the other hands, clients uses these resources.





Network definition → break the distinction between client and server: every peer can act as a client and/or as a server.

Is it a good definition?

Network definition: what about Skype?

Back in the days, Skype was branded as a P2P application. Why should Skype be P2P?

- Interaction between two peers?

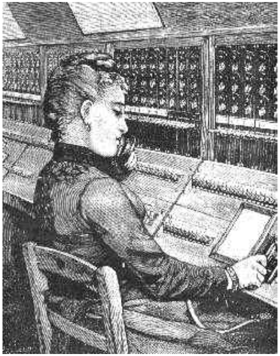
A very, very old tech indeed

Network definition: what about Skype?

Back in the days, Skype was branded as a P2P application. Why should Skype be P2P?

- Interaction between two peers?
- Interactions between any pair of peers?

An old tech



Network definition: what about Skype?

Back in the days, Skype was branded as a P2P application. Why should Skype be P2P?

- Interaction between two peers?
- Interactions between any pair of peers?
- From the makers of KaZaA and Joost (more to come later)?

Back to enumeration

Network definition: what about Skype?

Back in the days, Skype was branded as a P2P application. Why should Skype be P2P?

- Interaction between two peers?
- Interactions between any pair of peers?
- From the makers of KaZaA and Joost (more to come later)?
- Even the network definition is not perfect??

Alternate definitions: methodology, structure

A bit of fun: legal definition in France

P2P can be used for illegal stuff. The DADVSI law (2006) tries to forbid any «software obviously intended for the unauthorized making available to the public of protected works or objects».

A bit of fun: legal definition in France

P2P can be used for illegal stuff. The DADVSI law (2006) tries to forbid any «software obviously intended for the unauthorized making available to the public of protected works or objects».

- Covers P2P sharing (data and multimedia)
- Does not cover Skype (much)
- May cover most network apps (client/server included) : Apache, SSH. . .

A bit of fun: legal definition in France

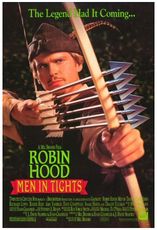
DADVSI hard to use in practice. Awareness of the difficulty of defining (legally) illegal P2P use.
Move to Hadopi (2009).

A bit of fun: legal definition in France

DADVSI hard to use in practice. Awareness of the difficulty of defining (legally) illegal P2P use. Move to Hadopi (2009).

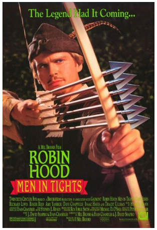
- Monitor target content (honeypot)
- Proving actual download by a physical person is impossible on a large scale
- Implementation of the characteristic negligence offence
- Technical failure, political success?

Methodology definition



P2P uses available resources (CPU, storage, bandwidth, content) wherever they are and redistributes them to those who need it.

Methodology definition

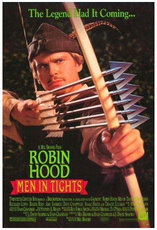


P2P uses available resources (CPU, storage, bandwidth, content) wherever they are and redistributes them to those who need it.

Two variants depending on what «wherever» means:

- At any user (covers BOINC)
- Anywhere in the network (CDN, EC2, . . .)

Methodology definition



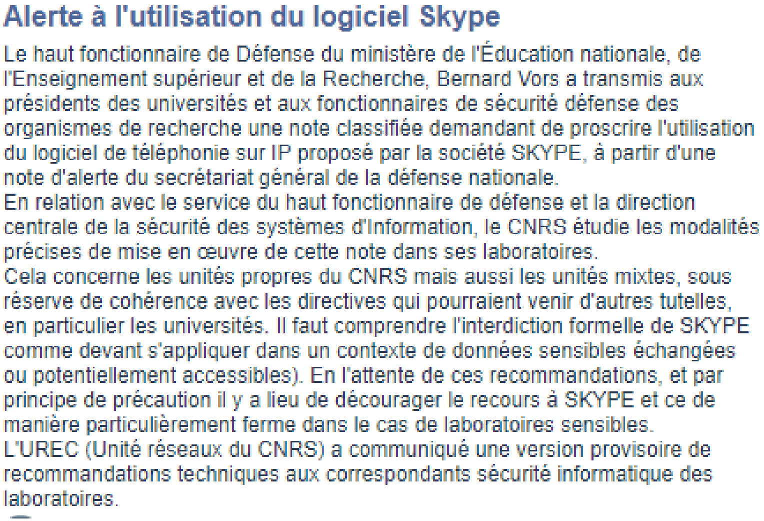
P2P uses available resources (CPU, storage, bandwidth, content) wherever they are and redistributes them to those who need it.

Two variants depending on what «wherever» means:

- At any user (covers BOINC)
- Anywhere in the network (CDN, EC2, . . .)

Skype: borderline

A bit of fun: Skype ban (2005)



Alerte à l'utilisation du logiciel Skype

Le haut fonctionnaire de Défense du ministère de l'Éducation nationale, de l'Enseignement supérieur et de la Recherche, Bernard Vors a transmis aux présidents des universités et aux fonctionnaires de sécurité défense des organismes de recherche une note classifiée demandant de proscrire l'utilisation du logiciel de téléphonie sur IP proposé par la société SKYPE, à partir d'une note d'alerte du secrétariat général de la défense nationale.
En relation avec le service du haut fonctionnaire de défense et la direction centrale de la sécurité des systèmes d'Information, le CNRS étudie les modalités précises de mise en œuvre de cette note dans ses laboratoires.
Cela concerne les unités propres du CNRS mais aussi les unités mixtes, sous réserve de cohérence avec les directives qui pourraient venir d'autres tutelles, en particulier les universités. Il faut comprendre l'interdiction formelle de SKYPE comme devant s'appliquer dans un contexte de données sensibles échangées ou potentiellement accessibles). En l'attente de ces recommandations, et par principe de précaution il y a lieu de décourager le recours à SKYPE et ce de manière particulièrement ferme dans le cas de laboratoires sensibles.
L'UREC (Unité réseaux du CNRS) a communiqué une version provisoire de recommandations techniques aux correspondants sécurité informatique des laboratoires.

Structural definition

Structure: how the elements of a system are arranged, and by extension, interact.
In networks, we can define by structure the answer to the question

Who gives what to whom?

Clients/servers, telephones, etc. generally have trivial structures.
Structural definition: P2P is characterized by non-trivial network structures.
Excludes trivial interactions (Skype?)

Definitions: take-away

Many definitions of P2P exist:

Enumeration	We all know what it is
Law	Mistake the “what” and the “how”
Network	No client/server distinction
Methodology	Tap unused resources
Structural	Non-trivial interactions

Recommendations

- There is no best answer (but there are bad ones)
- Use your brain!
- Is Skype P2P ? Who cares!

Definition for the course

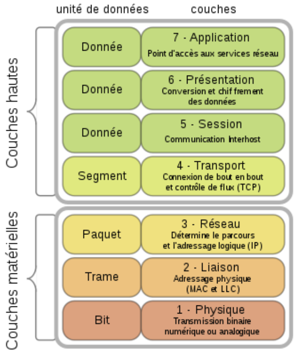
- Part III is mainly about (illegal?) content distribution
- Most non-stupid definitions are OK

Overlay

Re-re-reminder: OSI model

Network is split in 7 layers:

- Layer 7 (Application)
- Layer 4 (Point-to-point flow)
- Layer 3 (Routing)

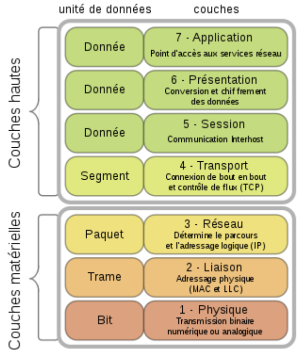

unité de données
couches
Couches hautes
Couches matérielles
Donnée
7 - Application
Point d'accès aux services réseau
Donnée
6 - Présentation
Conversion et chiffrement des données
Donnée
5 - Session
Communication Interhost
Segment
4 - Transport
Connexion de bout en bout et contrôle de flux (TCP)
Paquet
3 - Réseau
Détermine le parcours et l'adressage logique (IP)
Trame
2 - Liaison
Adressage physique (MAC et LLC)
Bit
1 - Physique
Transmission binaire numérique ou analogique

Overlay

Re-re-reminder: OSI model

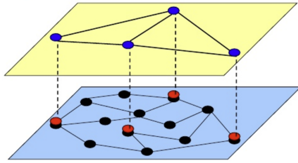
Network is split in 7 layers:

- Layer 7 (Application)
- Layer 4 (Point-to-point flow)
- Layer 3 (Routing)


unité de données
couches
Couches hautes
Couches matérielles
Donnée
7 - Application
Point d'accès aux services réseau
Donnée
6 - Présentation
Conversion et chiffrement des données
Donnée
5 - Session
Communication Interhost
Segment
4 - Transport
Connexion de bout en bout et contrôle de flux (TCP)
Paquet
3 - Réseau
Détermine le parcours et l'adressage logique (IP)
Trame
2 - Liaison
Adressage physique (MAC et LLC)
Bit
1 - Physique
Transmission binaire numérique ou analogique

P2P

- Creates a virtual network on top (overlay) of the physical network
- Overlay $\rightarrow$ graph
- Emulate routing / connectivity / . . . on the virtual topology
- A physical neighbor is not necessarily a virtual neighbor (and vice versa!)



Overlay: two main approaches

Structured graphs

Maintain a shape with good properties

- Performance is ensured by the graph structure
- Low maintenance for stable networks
- Fragile, high maintenance for dynamic networks
- Mostly unfit to heterogeneous networks

Unstructured graphs

Rely on randomness and self-adaptation

- Robust
- Adapted to heterogeneous networks
- Constant maintenance (costly on stable networks)
- No straightforward performance guarantee

Scalable

Client/server

- The capacity of servers is fixed
- Too many requests $\rightarrow$ DoS

P2P

- The capacity increases with the number of clients
- $\rightarrow$ P2P can in theory handle an arbitrary number of clients: scalability property

In practice, scalability is a real thing but it's not perfect

- Underlying physical network has limitations
- QoS constraints
- Size matters in practice (maintenance, logarithmic growth)

Fairness

Ideally

With P2P each peer can serve and be served by anyone: isotropic principle.

In practice

- It is not always possible to serve anything to anyone
- “All peers are equal, but some peers are more equal than others” (Orwell, the Server Farm)

$\rightarrow$ a peer is isotropic a priori when it enters the system

Resources and goals

Resources

- CPU
- Storage
- Content
- Bandwidth

Goals

- Distributed computed
- Content distribution
- Communication (Skype)
- Gaming

Resources and goals

Resources

- CPU
- Storage
- **Content**
- **Bandwidth**

Goals

- Distributed computed
- **Content distribution**
- Communication (Skype)
- Gaming

Content distribution

Three main branches

Filesharing the generic solution

- Primary goal: retrieve the content
- Secondary goals: availability, download time

Content distribution

Three main branches

On-demand Anything anywhen

- Catalog size (similar to filesharing availability)
- Minimize (start-up delay)
- Maximize quality

Content distribution

Three main branches

Live streaming

- Everyone wants to watch the same thing at the same time
- Minimize play-out-delay
- Maximize quality

Content distribution

Three main branches

Branch	FileSharing	On-demand	Live
Streaming?	No	Yes	Yes
Challenge	Minimal	Anything Anywhen	Small delay
Technical edge	Elastic traffic	Known content	Homogeneous needs
Technical difficulty	Minimal	Heterogeneous needs	Impossible anticipation

Indexation vs Distribution

All P2P content distribution systems must answer two questions:

Where is what I want?

- Build / maintain an index
- On-the-fly resolution
- Rely on a centralized solution

How do I get what I want?

- Just download from someone
- Be smart

Prehistory: communications

A few key dates

-200000?	Speech (network layer 1)
-30000?	Paintings (storage)
-7000?	Writing
-600→ -100	Long distance (horses/pigeons/semaphores)
1454	Gutenberg
1876	Phone
1969	Arpanet
1993	MP3
1999	DSL (in France), DivX 3.11

Prehistory: communications

A few key dates

-200000?	Speech (network layer 1)
-30000?	Paintings (storage)
-7000?	Writing
-600→ -100	Long distance (horses/pigeons/semaphores)
1454	Gutenberg
1876	Phone
1969	Arpanet
1993	**MP3**
1999	**DSL (in France), DivX 3.11**

A technical threshold is reached

- Around 2000, we can retrieve content in a reasonable time

	Hand	Print	Modem	DSL 512k	DSL 20M	FTTH
Book	2-3d	$\frac{1}{2}$h	1m	6s	0.15s	10ms
WAV	$\frac{1}{2}$y	1-2d	1h	6m	10s	0.5s
DVD	100y	1y	10d	1d	<1h	4m
MP3	20d	3h	6m	30s	1s	50ms
DivX	12y	40d	1j	3h	5m	30s

A technical threshold is reached

- Around 2000, we can retrieve content in a reasonable time
- A simple metric: can we “stream”?

	Hand	Print	Modem	DSL 512k	DSL 20M	FTTH
Book	No	Yes	Yes	Yes	Yes	Yes
WAV	No	No	No	Presque	Yes	Yes
DVD	No	No	No	No	Yes	Yes
MP3	No	No	Presque	Yes	Yes	Yes
DivX	No	No	No	Presque	Yes	Yes

A technical threshold is reached

Around 2000

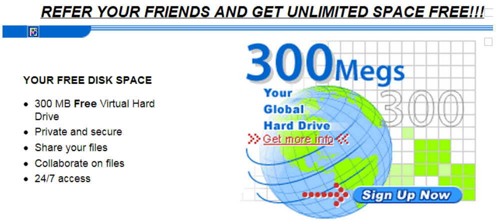
- Digital content becomes mainstream
  - CDs, MP3
  - DVDs, DeCSS, DivX
- High bandwidth enables download through Internet
- No YouTube, Netflix, Disney+, Prime, ...
- A few solutions for the geeks
  - Newsgroups (`alt.binaries.*`)
  - FTPz
  - IRC
  - Myspace

A technical threshold is reached

Around 2000

- Digital content becomes mainstream
  - CDs, MP3
  - DVDs, DeCSS, DivX
- High bandwidth enables download through Internet
- No YouTube, Netflix, Disney+, Prime, ...
- A few solutions for the geeks
  - Newsgroups (`alt.binaries.*`)
  - FTPz
  - IRC
  - **Myspace**

Intermezzo: Myspace, Dropbox before Dropbox


REFER YOUR FRIENDS AND GET UNLIMITED SPACE FREE!!!

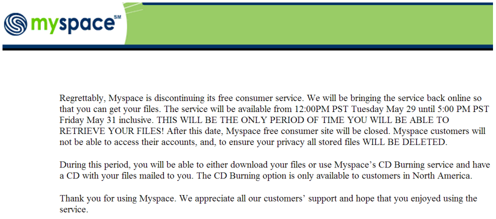
YOUR FREE DISK SPACE

- 300 MB **Free** Virtual Hard Drive
- Private and secure
- Share your files
- Collaborate on files
- 24/7 access

A success story

- MySpace offered 300MB «in the cloud»
- Unlimited account creation
- No Captcha
- $\rightarrow$ Creation of an unofficial API for automation
  - Split content in 300Mb chunks, create accounts, upload
  - Reconstruction infos spread through small files

Intermezzo: Myspace, Dropbox before Dropbox


Regrettably, Myspace is discontinuing its free consumer service. We will be bringing the service back online so that you can get your files. The service will be available from 12:00PM PST Tuesday May 29 until 5:00 PM PST Friday May 31 inclusive. THIS WILL BE THE ONLY PERIOD OF TIME YOU WILL BE ABLE TO RETRIEVE YOUR FILES! After this date, Myspace free consumer site will be closed. Myspace customers will not be able to access their accounts, and, to ensure your privacy all stored files WILL BE DELETED.

During this period, you will be able to either download your files or use Myspace's CD Burning service and have a CD with your files mailed to you. The CD Burning option is only available to customers in North America.

Thank you for using Myspace. We appreciate all our customers' support and hope that you enjoyed using the service.

The fall

- Income:
  - Ads: no money in Internet ads back then
  - Fremium model: didn't work for end users
- Costs:
  - Storage: expensive
  - Bandwidth: very, very, expensive
  - Quad damage from automation
- $\rightarrow$ business model was not sustainable

Intermezzo: Myspace, Dropbox before Dropbox

What can we learn from the MySpace story?

- Some people want to download (legal ?) content
- Too early ($\sim$10 years) for a viable commercial solution to emerge

A perfect setting

Circa 2000, the world is “P2P-ready”

- Means: plenty of downlink, untapped uplink
- Motive: retrieve content
- Opportunity: no competition

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	Gnutella: decentralized search
2000	KaZaA/eDonkey: hierarchical indexing
2003	BitTorrent: efficient distribution
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	PPLive/UUSee/TVAnts/. . . : P2P streaming

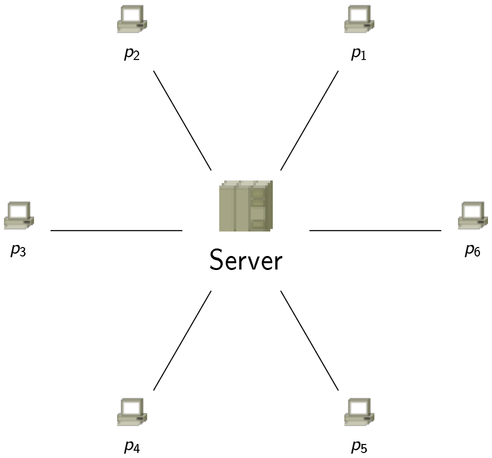
P2P: Key softwares

1998	**Napster: P2P content (MP3) distribution**
2000	Gnutella: decentralized search
2000	KaZaA/eDonkey: hierarchical indexing
2003	BitTorrent: efficient distribution
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	PPLive/UUSee/TVAnts/. . . : P2P streaming

Napster

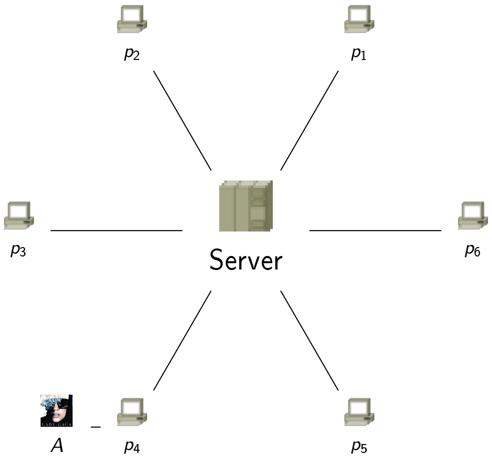
- Created in 1998 by Shawn "Napster" Fanning (The Italian Job) et Seam Parker (appears in The social network)
- Trial in 1999 (MPAA) - closed in July 2001
- Three steps system:
  1. Indexation: users upload the meta-data of their MP3 (catalog) to a server, `napster.com`
  2. Request: ask the server for some file (by name) ; receives a list of Napster clients (IP adresses) that claim hosting a compatible file
  3. Distribution: Download the MP3 from one client from the received list

Napster



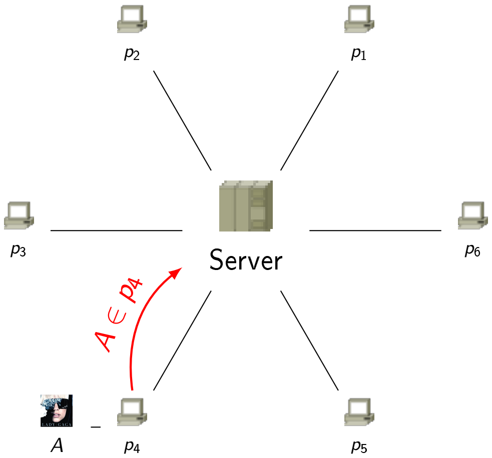
1. Indexation

Napster



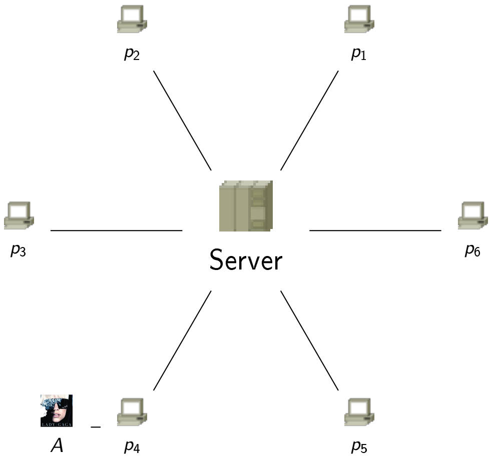
1. Indexation
   - p_4 owns A

Napster



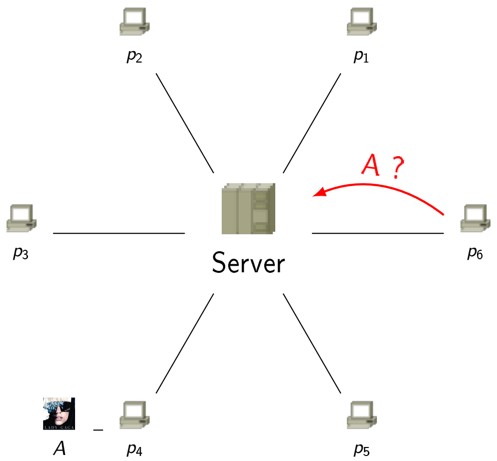
1. Indexation
   - p_4 owns A
   - p_4 notifies s

Napster



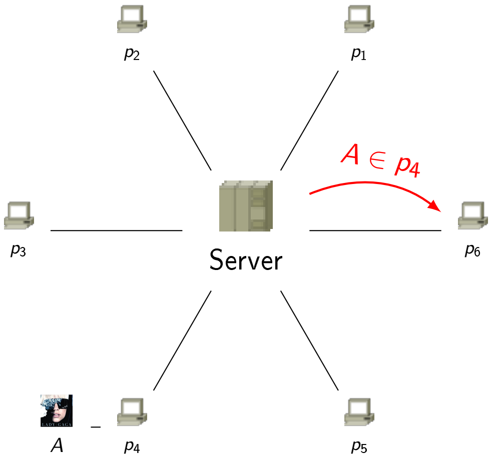
1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request

Napster



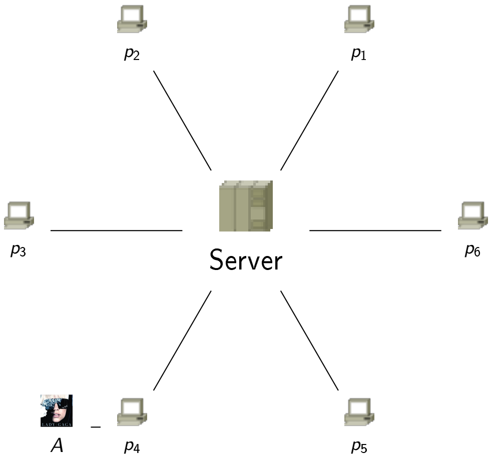
1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request
   - p_6 asks s who owns A

Napster



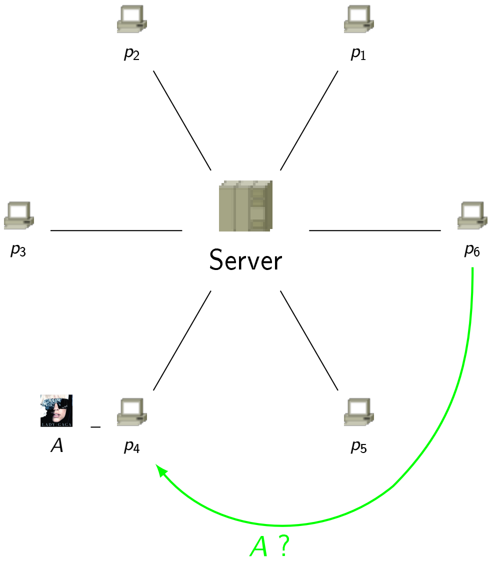
1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request
   - p_6 asks s who owns A
   - s responds p_4 (IP)

Napster



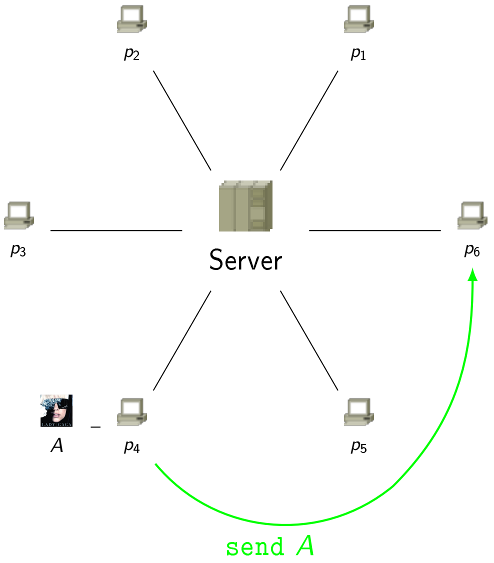
1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request
   - p_6 asks s who owns A
   - s responds p_4 (IP)
3. Distribution

Napster



1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request
   - p_6 asks s who owns A
   - s responds p_4 (IP)
3. Distribution
   - p_6 contacts p_4

Napster



1. Indexation
   - p_4 owns A
   - p_4 notifies s
2. Request
   - p_6 asks s who owns A
   - s responds p_4 (IP)
3. Distribution
   - p_6 contacts p_4
   - p_4 sends to p_6

Napster

Innovation

- Central indexation $\rightarrow$ complex requests (SQL queries, regex) are feasible
- Distribution from peer to peer (literally)

Limits

- Central index $\rightarrow$ single point of failure (indeed…)
- “Stupid” distribution (one-to-one, whole file)

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	**Gnutella: decentralized search**
2000	KaZaA/eDonkey: hierarchical indexing
2003	BitTorrent: efficient distribution
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	PPLive/UUSee/TVAnts/. . . : P2P streaming

Gnutella

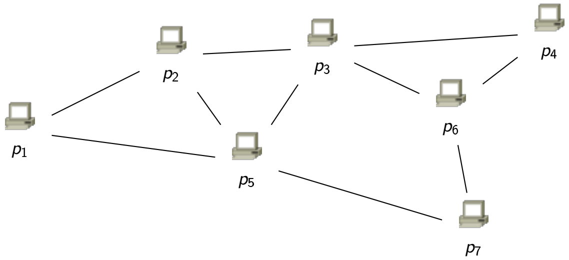
Histoiry

- 1st client : 2000 (nullsoft-AOL). Pulled-off after 1 day (!)
- Code available → open-source clones (LimeWire, Morpheus...)
- Biggest IP of the year in its time (quasi-extinct today)

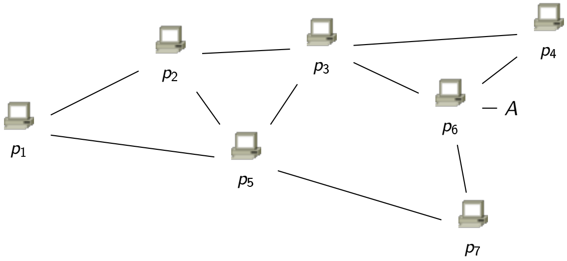
Protocol idea (0.4)

- Fully decentralized: no central index
- Searches are made by **flooding**
- Many enhancements afterwards (hash, ultrapeers, chunks...)

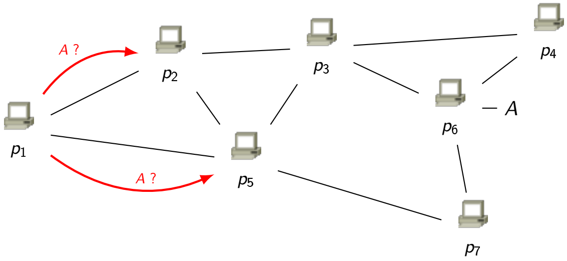
Gnutella



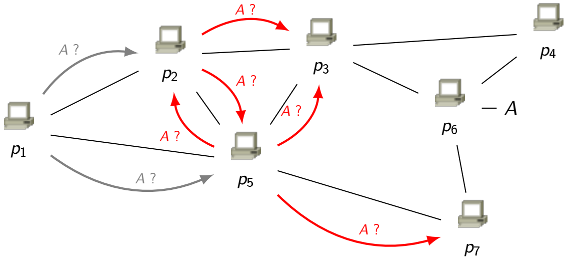
Gnutella



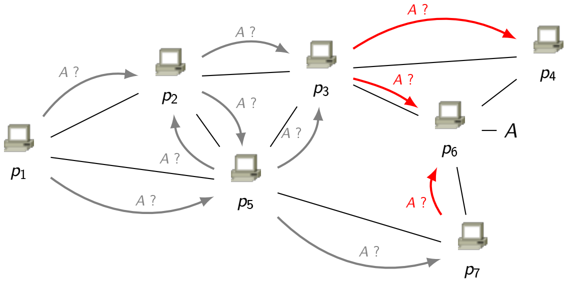
Gnutella



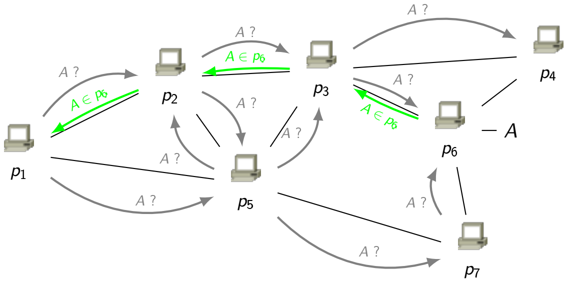
Gnutella



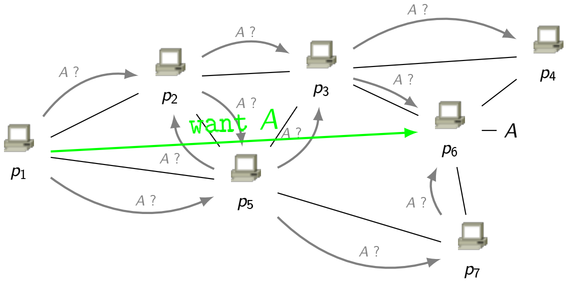
Gnutella



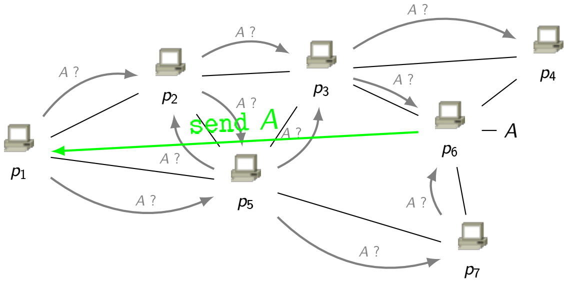
Gnutella



Gnutella



Gnutella



Gnutella

Innovation

No index, regex still feasible

Limits

- Flooding is BAD: for every request, only a fraction of the peers receives query
  - Too small fraction: partial response
  - Too large fraction: overhead (measured proportion of request messages: more than half traffic)
- Distribution is still “stupide”

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	Gnutella: decentralized search
2000	**KaZaA/eDonkey: hierarchical indexing**
2003	BitTorrent: efficient distribution
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	PPLive/UUSee/TVAnts/. . .: P2P streaming

KaZaA

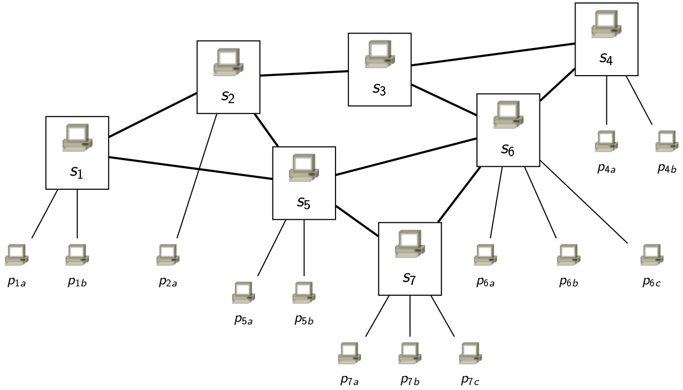
History

- 2000, Niklas Zennström et Janus Friis (Skype, Joost)
- Relies on the FastTrack overlay network (proprietary)
- Biggest traffic of the year, extinction

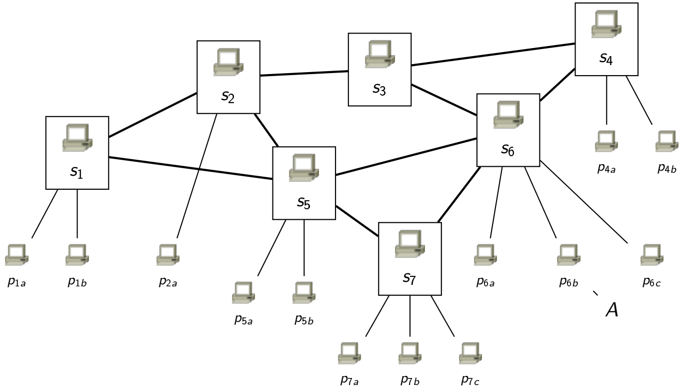
Key idea: **hierarchical index**

- Supernodes: peers with high bandwidth/availability
- Each ordinary node is attached to one supernode
- $\sim 100 - 1000$ ordinary nodes for one supernode
- Supernodes manage their nodes (indexation, requests) and communicate through flooding (Gnutella-style)
- Also, chunk-based distribution that relies on a reputation mechanism

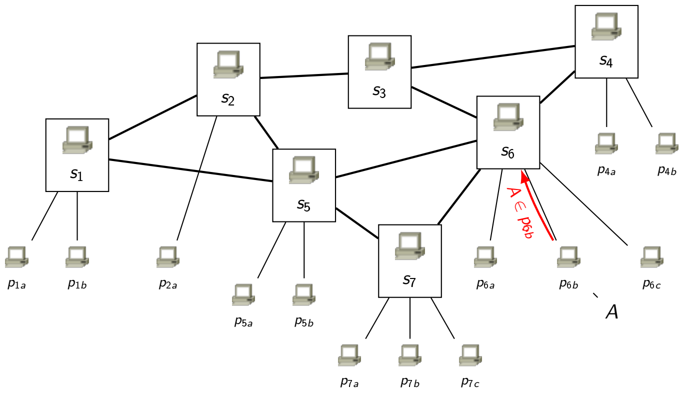
KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{4a}
p_{4b}
p_{5a}
p_{5b}
p_{6a}
p_{6b}
p_{6c}
p_{7a}
p_{7b}
p_{7c}

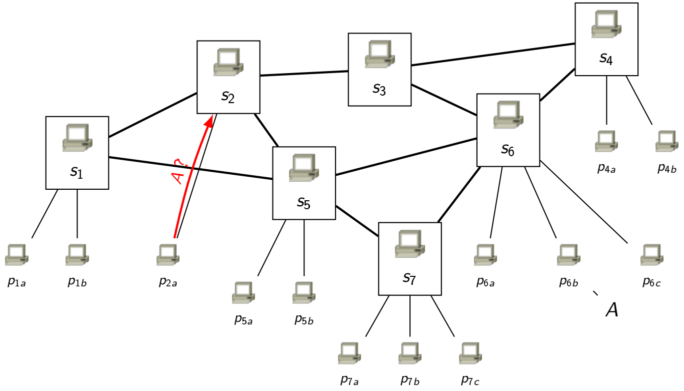
KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{5a}
p_{5b}
p_{7a}
p_{7b}
p_{7c}
p_{6a}
p_{6b}
p_{6c}
p_{4a}
p_{4b}
A

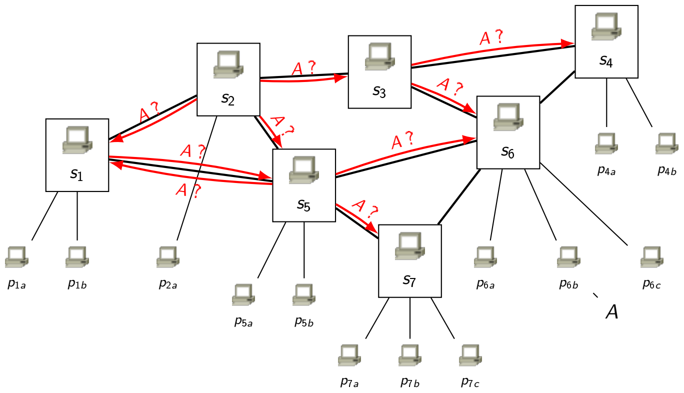
KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{5a}
p_{5b}
p_{7a}
p_{7b}
p_{7c}
p_{6a}
p_{6b}
p_{6c}
p_{4a}
p_{4b}
$A \in p_{6b}$
A

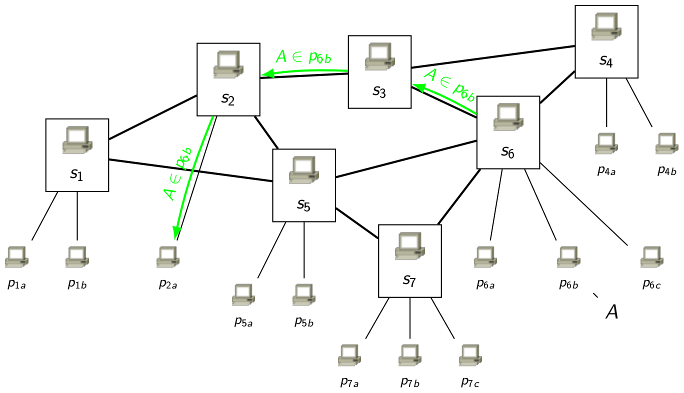
KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
A?
p_{5a}
p_{5b}
p_{7a}
p_{7b}
p_{7c}
p_{6a}
p_{6b}
p_{6c}
A
p_{4a}
p_{4b}

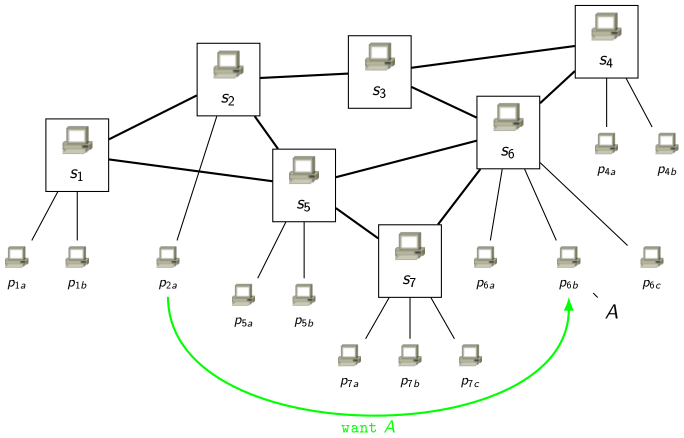
KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
A ?
p_{1a}
p_{1b}
p_{2a}
p_{4a}
p_{4b}
p_{5a}
p_{5b}
p_{6a}
p_{6b}
p_{6c}
p_{7a}
p_{7b}
p_{7c}
A

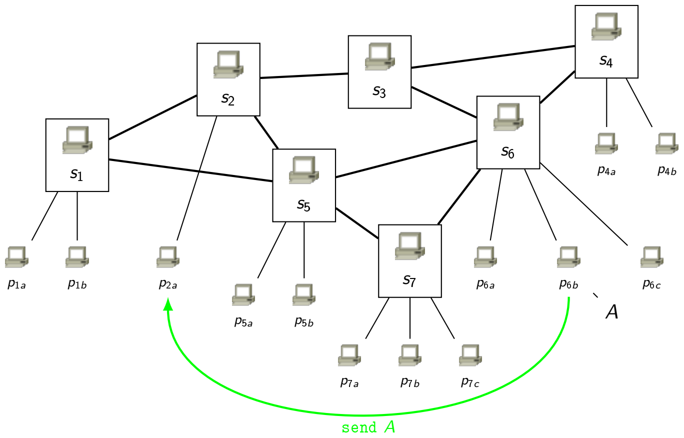
KaZaA


$A \in p_{6b}$
$A \in p_{6b}$
$A \in p_{6b}$
s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{4a}
p_{4b}
p_{5a}
p_{5b}
p_{6a}
p_{6b}
p_{6c}
A
p_{7a}
p_{7b}
p_{7c}

KaZaA


s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{5a}
p_{5b}
p_{7a}
p_{7b}
p_{7c}
p_{6a}
p_{6b}
p_{6c}
p_{4a}
p_{4b}
A
want A

KaZaA



s_1
s_2
s_3
s_4
s_5
s_6
s_7
p_{1a}
p_{1b}
p_{2a}
p_{5a}
p_{5b}
p_{7a}
p_{7b}
p_{7c}
p_{6a}
p_{6b}
p_{6c}
p_{4a}
p_{4b}
A
send A

eMule - eDonkey

History

- eDonkey2000 client released in. . . 2000
- MetaMachine corp., closed in 2005 (RIAA)
- Documented protocol $\rightarrow$ open-source clones (eMule)
- Biggest traffic. . . (still active today)

Principle

- eDonkey servers: machines dedicated to indexation (“professional” supernodes)
- Chunk-based distribution that relies on a queuing system

KaZaA/eDonkey

Innovation

- Hierarchical indexing
- Chunk (overlay data “packet”)

Limitations

- KaZaA: sequential chunk scheduler $\rightarrow$ “missing chunk” syndrome
- KaZaA Lite K++: fake reputation, always supernodes
- eDonkey: Saturation of queues $\rightarrow$ very large delays
- Servers / supernodes $\rightarrow$ points of failures

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	Gnutella: decentralized search
2000	KaZaA/eDonkey: hierarchical indexing
2003	BitTorrent: efficient distribution
2006	**R.I.P. Razorback 2.0 and rise of DHTs**
2006-2009	PPLive/UUSee/TVAnts/...: P2P streaming

KAD vs Razorback 2.0

Kad

RazorBack 2.0

The circle of life

KAD vs Razorback 2.0

Kad

- Goal: avoid (semi-)centralized indexing AND flooding
- Solution: Distributed Hash Tables, introduced in 2001 (research papers)
- The index is distributed over all peers
- One DHT, Kad, is implemented in eMule quite quickly

RazorBack 2.0

The circle of life

KAD vs Razorback 2.0

Kad

RazorBack 2.0

- A race for the best eDonkey server begins in the early 2000s
- One clear winner: Razorback 2.0
- Other servers become useless (spam, small catalog and low reactivity)

The circle of life

KAD vs Razorback 2.0

Kad

RazorBack 2.0

The circle of life

- At start, Kad is not popular (R2 is much faster)
- February 21, 2006, forced shutdown of Razorback2.X
- Lots of users switch to Kad, the only viable alternative...
- ...and **it works!** (supports the load explosion)

Distributed Hash Tables

Innovation

Fully decentralized index

Limitation

- Only simple combinations of exact requests (~~regexp~~)
- Logarithmic price (delay and maintenance)

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	Gnutella: decentralized search
2000	KaZaA/eDonkey: hierarchical indexing
2003	**BitTorrent: efficient distribution**
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	PPLive/UUSee/TVAnts/. . . : P2P streaming

BitTorrent

History

- Founding paper and client proposed by Bram Cohen in 2003
- Documented protocol
- Lots of implementations (vanilla, Azureus/Vuze, μtorrent. . .)
- Biggest traffic (by years) up to Youtube take-over. Still biggest P2P traffic

Key idea: focus on the content to distribute

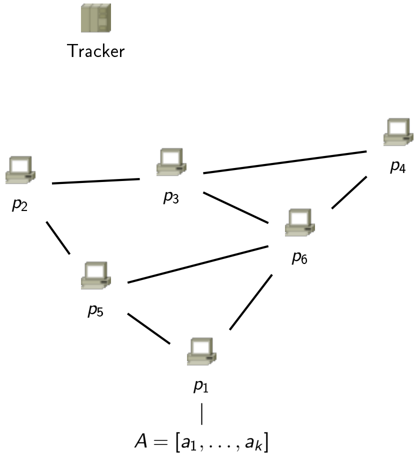
- One dedicated overlay per content: the swarms
- A swarm is managed by a tracker
- Tit-for-Tat incentive policy: Give and you shall receive
- + 2/3 others simple yet efficient tricks

BitTorrent

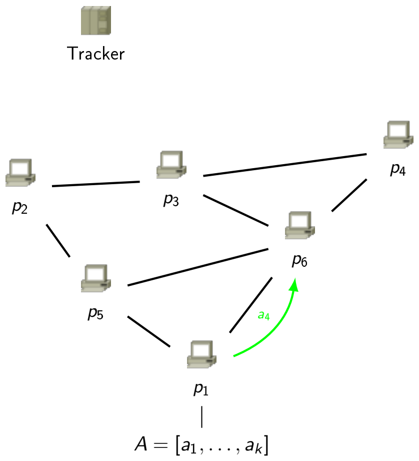
Protocol overview

- The tracker monitors participating peers
- On arrival, a peer receives neighbors
- Chunks are exchanged until download completion

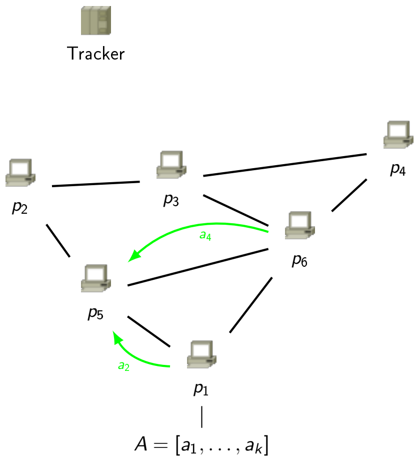
BitTorrent



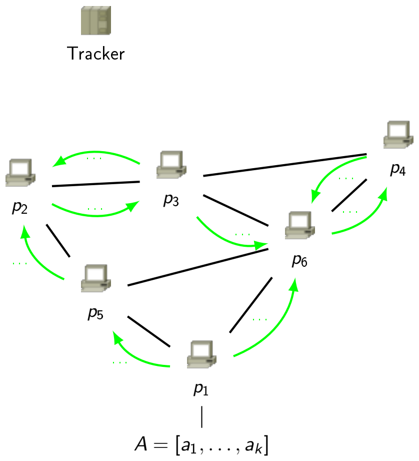
BitTorrent



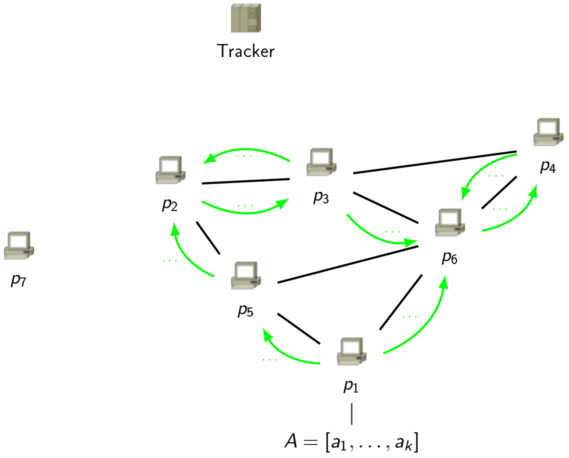
BitTorrent



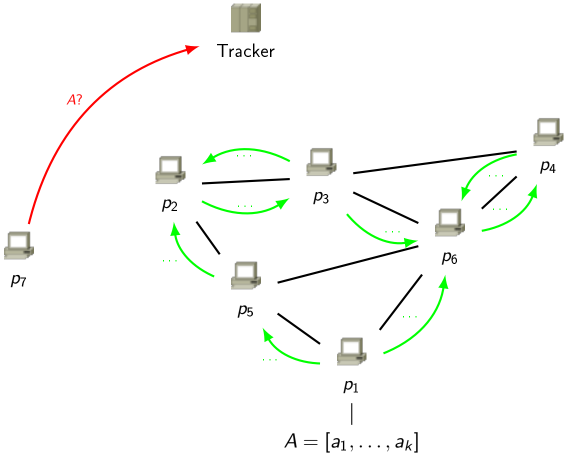
BitTorrent



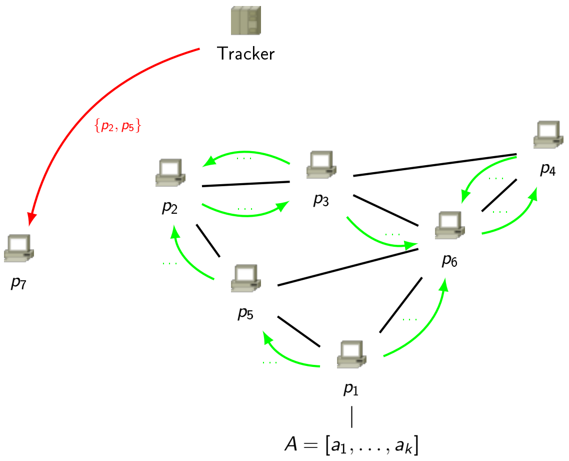
BitTorrent



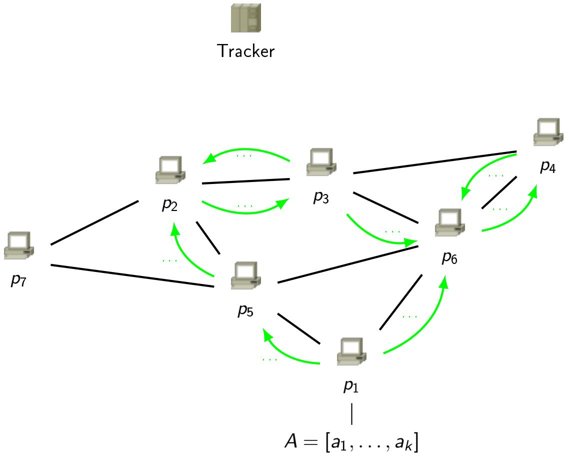
BitTorrent



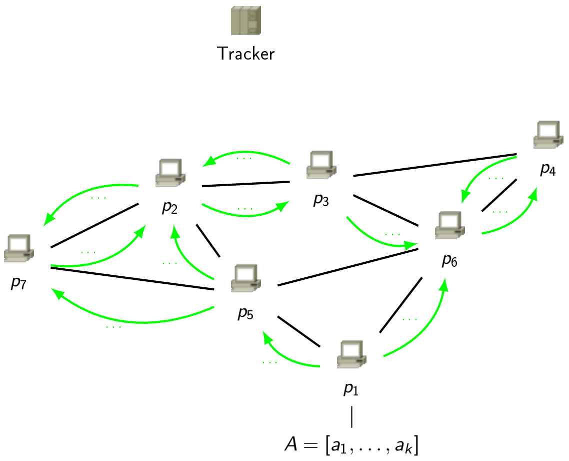
BitTorrent



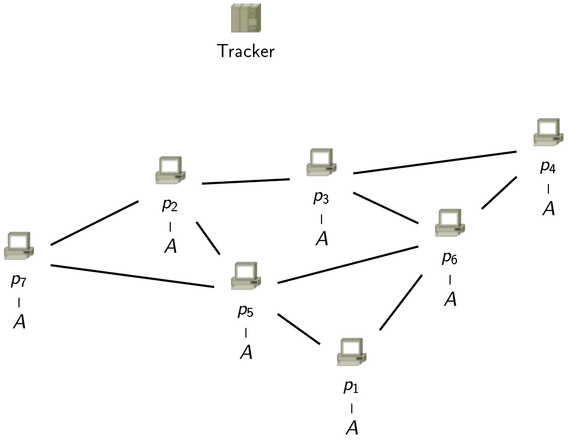
BitTorrent



BitTorrent



BitTorrent



BitTorrent

Innovation

- Content-centric instead of user-centric
- Top distribution algorithms

Limitations

- Trackers $\rightarrow$ points of failures
- Swarm lifespan is smaller than eMule retention

P2P: Key softwares

1998	Napster: P2P content (MP3) distribution
2000	Gnutella: decentralized search
2000	KaZaA/eDonkey: hierarchical indexing
2003	BitTorrent: efficient distribution
2006	R.I.P. Razorback 2.0 and rise of DHTs
2006-2009	**PPLive/UUSee/TVAnts/. . . : P2P streaming**

P2PTV

History

- Goal: sport events
- Many proprietary incompatible solutions (UUSee, PPLive, TVAnts,. . .)
- Growth killed by YouTube and CDN-like solutions
- Mostly used today in Asia and “expat” communities

Idea: lower the delay

- Minimize buffer length
- Optimize chunk diffusion
- Boost the system with actual servers

P2PTV

Innovation

Algorithms for delay optimization

Limitations

- The logarithmic price
- Youtube, Akamai, Netflix

Part III

B) Distributed Hash Tables (DHTs)

Distributed Hash Tables

Motivation

Find where a content is at low cost

Problems to solve

Decentralisation	the burden must be shared among all peers
Scalability	search must work on very large systems
Dynamicity	peers arrive and leave, nicely. . . or not!

Distributed Hash Tables

Hash function

Function $h : E \mapsto F$
E = anything ! (file contents, file names, IP addresses. . .)
F = key space (circle $[0, ..2^n[$, square, hypercube. . .)

Principle of a DHT (Distributed Hash Table)

- Each peer has an identifier (key / hash)
- Each data has an identifier (key / hash)
- Each peer is in charge of an area of F (things whose key is inside the area)
- Find a key $\rightsquigarrow$ find a route to the peer in charge of that key
- Lots of variants depending on F and the routing mechanism
  - CAN, Pastry/Tapestry, Chord,. . .

One important thing

Managing a key does not mean possessing the content!

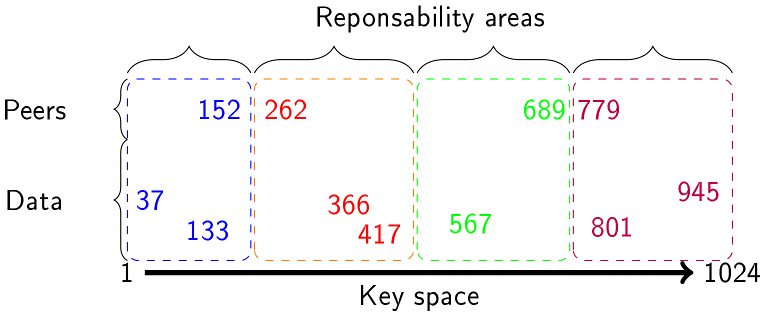
One important thing

Managing a key does not mean possessing the content!

Managing a key is knowing:

- what corresponds to the key
- associated meta-data (IP addresses, . . .)

Toy example: nearest neighbor indexation



Peer with key 152 manages data with keys 37 and 133...

Request example: “House of the Dragon”

1. h(“House”) = 1234
2. Search the peer nearest to 1234
3. It is the peer with hash 1100
4. Ask for a list of contents
5. h(“Dragon”) = 5678
6. Nearest peer: 5600 $\rightarrow$ request $\rightarrow$ contents
7. Intersection of contents
8. Display
9. User chooses content with hash 6666
10. Peer with hash nearest to 6666: 6667
11. Ask list of IPs to 6667
12. Download from the IPs

Routing by key

Principle of the routing algorithm

- Peer P looks for data with key c
- Data is indexed at peer Q
- P knows a few neighbors
- P transmits the request to the neighbor that is «closest» to c
- The neighbor re-transmits, etc.
- Until Q is reached

Performance

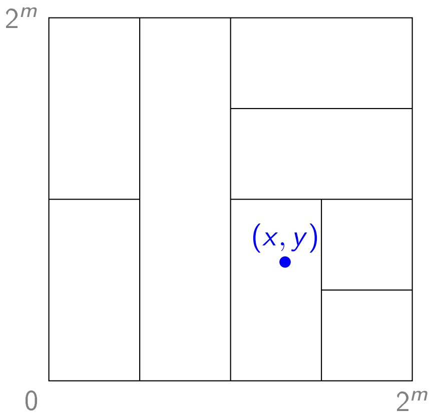
- We should try to make it in $O(\log n)$ forwards (jumps)
- Looks a bit like dichotomic search
- No more flooding!

Routing by key

How to do that?

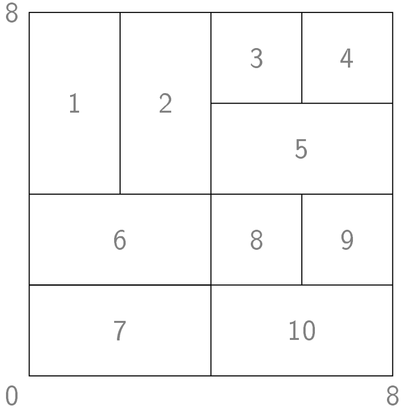
- The overlay tries to have some **topology** :
  - CAN : multi-dimensional torus
  - Chord : circle (with... chords)
  - Viceroy : butterfly graph
  - Tapestry/Pastry/Kademlia : hypercube
  - D2B : de Bruijn graph
- We use the topology to navigate the graph
  - From neighbor to neighbor
  - At each step we get closer to destination
- If the graph diameter is $O(\log n)$...
- ... We should reach destination in $O(\log n)$ steps!

Content-Addressable Network (CAN)

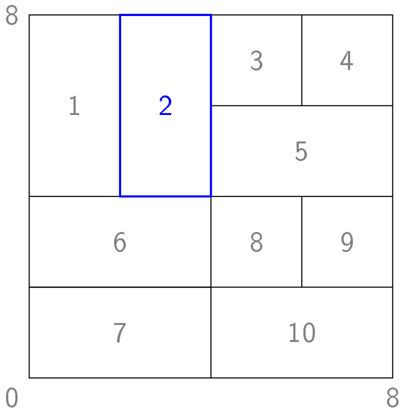


- F : hypercube with d dimensions ($d = 2$)
- Each peer manages a rectangle
- d hash functions $\rightsquigarrow$ each key is d-uplet (a pair) of hashs
- Routing to the adjacent neighbor of closest area for Manhattan distance

Example



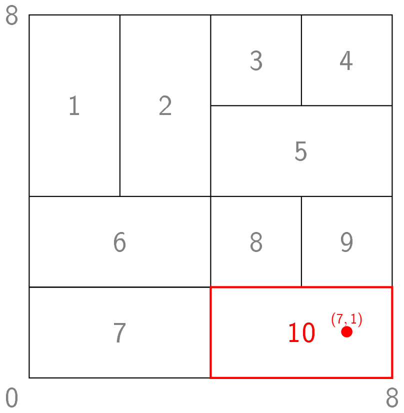
Example



Routing table of 2

Peer	Area
2	(2, 4) à (4, 8)
1	(0, 4) à (2, 8)
3	(4, 6) à (6, 8)
5	(4, 4) à (8, 6)
6	(0, 2) à (4, 4)

Example

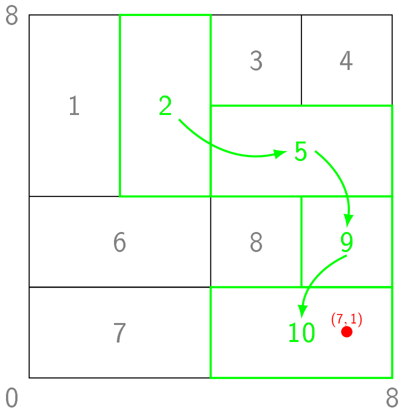


Routing table of 2

Peer	Area
2	$(2,4)$ à $(4,8)$
1	$(0,4)$ à $(2,8)$
3	$(4,6)$ à $(6,8)$
5	$(4,4)$ à $(8,6)$
6	$(0,2)$ à $(4,4)$

- Peer 2 looks for key $(7,1)$, possessed by peer 10.

Example



Routing table of 2

Peer	Area
2	$(2,4)$ à $(4,8)$
1	$(0,4)$ à $(2,8)$
3	$(4,6)$ à $(6,8)$
5	$(4,4)$ à $(8,6)$
6	$(0,2)$ à $(4,4)$

- Peer 2 looks for key $(7,1)$, possessed by peer 10.
- Route from 2 to $(7,1)$: $2 \rightarrow 5 \rightarrow 9 \rightarrow 10$.

Arrivals and departures

- Arrival of A : A takes a key (X, Y) and contacts an entry peer B (bootstrap)
  - A contacts through B the peer C in charge of (X, Y)
  - C splits its area with A (they adjust their routing tables)
  - A and C contact the neighbors for updating their tables
- Departure of A : A determines who should take over and contact its neighbors
  - All neighbors are notified / updated
  - The area may be dispatch among several neighbors

Properties

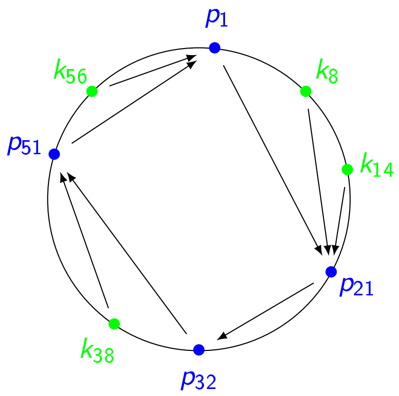
- Easy to understand, easy to extend
  - Overlapping areas for more robustness
  - Dimension d can be larger
  - Dynamic reshaping of areas to balance the load
- Size of routing table: $O(d)$ ☺
- Number of jumps: $O(dn^{1/d})$ ☹
- In the end: useless (costly in messages, slow)

The Chord protocol

- Always the same recipe...
- Hash function: SHA-1
- $F = [0, ..2^m[$ $(m = 160)$ seen as a circle modulo 2^m: the Chord ring
- Routing building block: successor(k) = first peer with key strictly greater than k modulo 2^m

Successors

A peer manages all the keys of which it is the successor

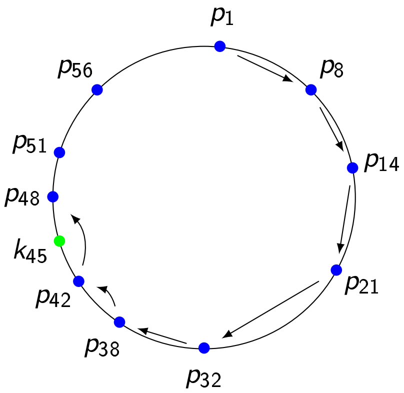


The maths corner

With high probability, for N peers and K keys:

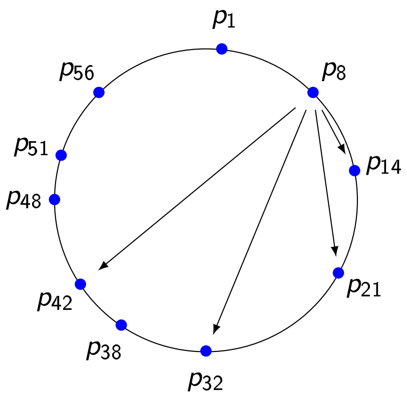
- Each peer manages at most $\frac{(1+C)K}{N}$ keys
- $C = O(\ln(N))$
- Is a peer arrives or leaves, at most $O(\frac{K}{N})$ keys need to change successor

Successor routing



- Example: 1 looks 45 (managed by 48)
- ☹☹☹

Solution : finger table

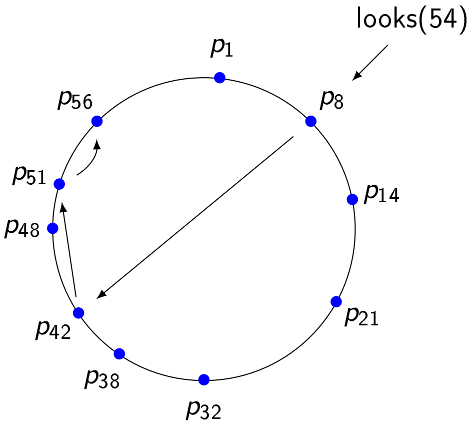


- finger(k) : successor of $n + 2^{k-1}$ modulo 2^m

Finger table of p_8

$S(8 + \{2^0, 2^1, 2^2\})$	p_{14}
$S(8 + \{2^3\})$	p_{21}
$S(8 + \{2^4\})$	p_{32}
$S(8 + \{2^5\})$	p_{42}

Finger table routing



A fast routing

- Each jump halves the distance (more or less)
- Theorem: with high probability, the number of jumps to locate the successor of a key is $O(\ln(N))$
- Theorem: with high probability, the number of entries in the finger table is $O(\ln(N))$

Departure and arrivals

- On arrival, a peer N contacts an entry point A (bootstrap)
  - A looks B, predecessor of N
  - N becomes the successor of B
  - N contacts B, retrieve the table
  - N contacts the table so all can adjust entries
  - N contacts its successor to share the keys
- On departure, a peer contacts its predecessors (finger included) to notify the leave. Keys are sent to the successor. Predecessor is connected to successor.

Chord properties

- Correct even if finger are corrupted (worst case: successor routing)
- Redundancy feasible to avoid data loss (e.g. host copy of the keys of predecessor and successor)
- Possibility to cache request results to accelerate future requests
- Relatively easy to understand (yes, yes!)

DHT: conclusions

Existing topologies

- Chord [Stoica & alii, 2001] : circle
- CAN [Ratnasamy et al., 2001] : multidimensional torus
- Pastry [Rowstron et Druschel 2001] : hypercube
- Viceroy [Malkhi et al., 2002] : butterfly
- Kademlia [Maymounkov Mazières 2002] : hypercube
- D2B [Fraigniaud et Gauron, 2003] : de Bruijn

DHT: take-away

Benefits

- Fast routing: $O(\log N)$, possibly $O(\frac{\ln N}{\ln \ln N})$ ($b = \ln N$ on the hypercube)
- Compact routing tables: $O(\log N)$, $O(\frac{(\ln N)^2}{\ln \ln N})$, or even constant.
- Much harder to destroy than a single server
- True P2P, true scalability

Issues

- Anything goes through the hash function $\rightarrow$ no complex request !
- Slower than a traditional server
- Protocols can be complex, bugs are hard to address

Acknowledgments

Part III owns a lot to Laurent Viennot and Fabien de Montgolfier.