Introduction on Peer to Peer systems

Georges Da Costa

Yerevan, Armenian National Academy of Sciences

Goal of this Lecture

- What can P2P do, not only as a buzzword
- What it can't do
- Shows some examples & algorithms

A Survey and Comparison of Peer-to-Peer Overlay Network Schemes, by Eng Keong Lua and al.
in *IEEE Communications survey and tutorial* March 2004

Harnessing the Power of Disruptive Technologies
published by O'Reilly, 2001

1 What is P2P

2 First generation systems

3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

Plan

1 What is P2P

2 First generation systems

3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

Universal

What have in common

- Net Meeting, Skype, Ekiga
- Irc, Msn, Icq, Jabber
- Kazza, Freenet, Napster, Gnutella
- Seti@Home, Folding@Home
- Ebay, Flickr, Facebook

Definition

Philosophical one

Participants gathering their resources in order to achieve a common goal

Why ?

Available resources

- Large Hard Drives
- Powerful CPUs
- Correct connexion to Internet

Users want

- More freedom
- No link to commercial companies
- No infrastructure cost

A new (?) solution : Peer to Peer systems

Definition

Participant gathering their resources in order to achieve a common goal

- Computers are running the same code
- There is no global view of the system
- View is limited to neighboors
- Everyone has the same rights and duties

Peer-to-Peer: New name, old concept

An architecture already there

- Internet connects most of existing computers
- Most computers are not fully used
  - Idle time > 75% on personal computers
  - Storage systems are mostly empty

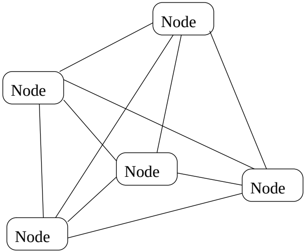
Already used between servers

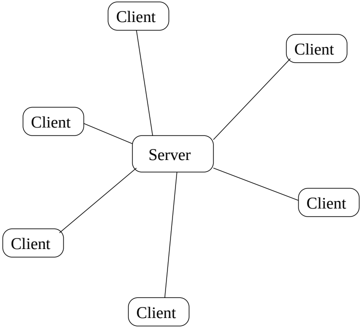
- Usenet
- DNS
- IP Routing

Comparison with Client/Server

- In client/Server each node is either a Client or a Server. Usually there are a few Servers and lots of Clients.
- Client/Server systems suffer from *single point of failure*.
- Client/Server are mostly static, at least the Servers. Peer to Peer systems are dynamics.
- Client/Server systems need *human* administrators
- Client/Server does not scale

Comparison with Client/Server II





Comparison with Client/Server II

When a new participant joins a service, the service increase the resource consumption

- **Client/Server** : increases the server power/connectivity
- **Peer to Peer** : uses the resources given by the participant

Not so easy

Wanted

- Scalability (1K,100K,1M nodes)
- Dynamicity
- Security (user, task)
- Transparent
  - For the user (CPU,memory,disk)
  - For the network
- Heterogeneity
- Self-organization
- Participation (66% of Free riders)
- Go through NAT/Firewall

Self-organization

Participants

High volatility & voluntary

- No central administration
- Resource discovery
- Heterogeneity
  - Hardware
  - Users (15% of users have 94% of files)
- Distribution of the resources
- Trust

What's not new

Partial solutions

- Scalability : Farm of web servers
- Dynamism : Cell phones
- Fault tolerance : Redundant servers

Current Peer to Peer systems

Available applications

- *File sharing*
- Distributed storage
- Content delivery
- Distributed computing
- Telephony/Chat
- Games

Current Peer to Peer systems (cont)

Widely used

2004 : According to British Web analysis firm CacheLogic, BitTorrent accounts for an astounding 35 percent of all the traffic on the Internet – more than all other peer-to-peer programs combined – and dwarfs mainstream traffic like Web pages

Start-ups

- Skype (ok, no more a small start-up)
- BitTorrent
- UbiStorage

Two worlds

Internet Users

- Problem of security
- Large scale
- No control
- Motivation needed

Private Area (Corp., Univ.)

- Other mean of security
- Medium to large scale
- Total control

Plan

1 What is P2P

2 First generation systems

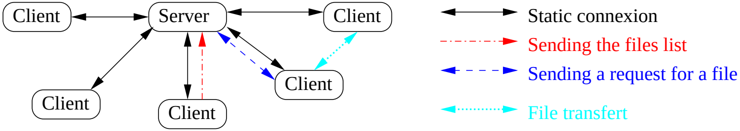
3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

Index Method



- Users send the list of their files to a server
- To find a file, you send a request to the server
- It answers with the list of clients owning the file
- You directly contact the owners for the transfer

Index Method II

Systems

Napster, Mojonation, Yaga, Filetopia, Seti@Home

Problems

- Scaling
- Price
- HotSpot
- Attack
- Single point of failure

Useful when...

- Small number of client
- Need a total control of transfers (video game industry)
- Performance is more important than cost

BitTorrent

Same approach as Napster, but :

- Downloads are done in parallel
- One server per file
- Server manages all the details of transfers
- Server enforces the rule *The more you share, the more you get*

Differences

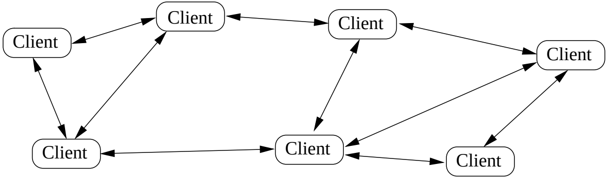
- Specialized for large files
- Distributed due to the *One server per file* rule

Privacy

No privacy

- Napster : The server knows all transfers
- BitTorrent : For each file, a server knows all transferts

Flooding



- You send your request to your neighbors
- They forward it to their neighbors, and so on until reaching the *Time To Live* depth
- Users with files corresponding to the request answer

Flooding II

Systems

Gnutella, Direct Connect

Characteristics

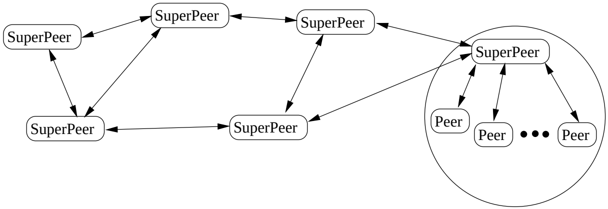
- Distributed structure
  - No single point of failure
  - Denial of service difficult (but possible)
- Not scalable
  - Resource consumption (network)
  - Not complete answers

Privacy

Average to good privacy

- Onion routing (good privacy)
- No global view of the system
- Usually easy to obtain the shared list of a node
- Difficult to have a global impact

Super Peers



Super Peers act as local servers

- Some reliable nodes act as super peers
- Super peers are connected with a gnutella protocol
- Each super peer acts as a local server for several peers

Super Peers II

Systems

Gnutella2, Kazaa

Characteristics

- Less distributed structure
    - Some nodes are more loaded
    - Some nodes are more important
- Scalable
    - Less resource consumption due to limits of number of answers

Plan

1 What is P2P

2 First generation systems

3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

A case study : Freenet

Ian Clarke, University of Edinbourgh, (1999)

Keywords

- A peer-to-peer file sharing system
- Provide anonymity for authors and readers
- A web of Freedom

Principle

- Files are referenced by key
- The key is obtained by SHA-1 on the file
- The key is routed to localize the file

Content Driven routing algorithm

- Routing table contains a set of key/node pairs
- Take the nearest key in the routing table to obtain the next node to consult.
- Nearest key = by lexical comparison

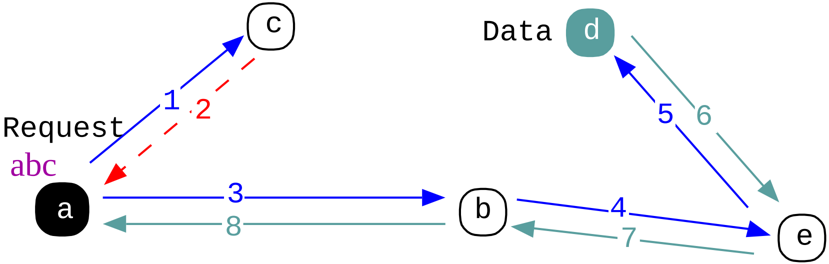


abb	node c
acd	node b

On the path of the answer

- File is replicated on the path in the cache
- Cache : variant of *L*ast *R*ecently *U*sed
- Routing tables are updated

→ the graph evolves (new links = new entries)

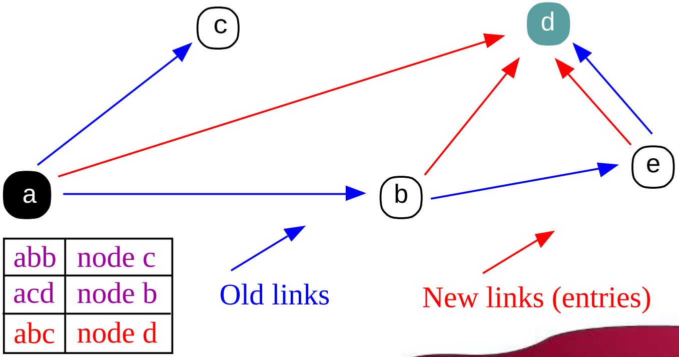


abb	node c
acd	node b
abc	node d

Anonymity

Reader

- Impossible to know if a user is forwarding or initiating the request
- Impossible to know if a user is the last to receive a file

Writer

- Once in the system, the writer can disconnect
- Impossible to know if someone insert some file or forward it

Some properties

Self-organization of the graph

- Nodes specialize in files with close keys (learning process)
- Good properties (Small World)

File are automatically replicated in function of their popularity

- Hot-spots are limited
- Tolerant against attacks

Drawbacks

Counterpart

- Files might disappear (LRU cache)
- The network is heavily loaded
- Difficult to update a value
- Impossible to know what is hosted locally

Plan

1 What is P2P

2 First generation systems

3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

Pastry

Principle

- Each file has a key
- Each node has an identifier
- Node with identifier *Id* manages keys whose values are near *Id*

Queries

- Content driven queries
- Suffix forwarding

Pastry II

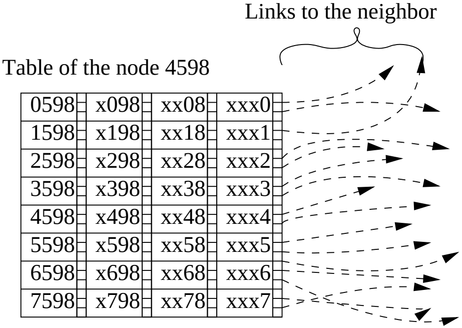
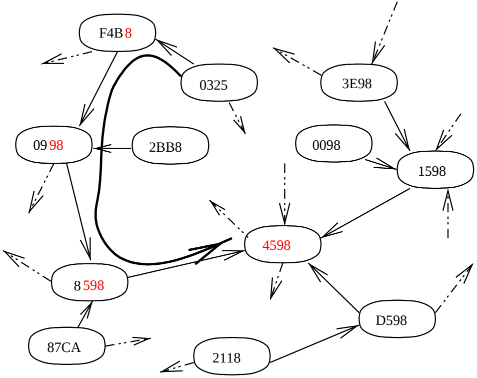


Table of the node 4598

Links to the neighbor

0598	x098	xx08	xxx0
1598	x198	xx18	xxx1
2598	x298	xx28	xxx2
3598	x398	xx38	xxx3
4598	x498	xx48	xxx4
5598	x598	xx58	xxx5
6598	x698	xx68	xxx6
7598	x798	xx78	xxx7



- Neighbors of *Id* are chosen as to have the suffix of their identifier in common with *Id*

Pastry III

Pros

- $ln(n)$ messages guarantee
- Good path redundancy

Cons

- Difficult to keep a synchronized neighbor table
- Problem of data redundancy
- No adaptation to data dynamicity

Plan

1 What is P2P

2 First generation systems

3 Self-organized systems

4 Structured systems

5 Distributed Hash Table

6 Conclusion

Current state of Peer to Peer systems

- A lot of redundant systems
- Typically File Sharing

Common basic component

Distributed index (Key, Value)

- Key is typically the filename
- Value is typically the file content or where to obtain it

Each Key is associated with a node

Generic Interface

- Node Id : k-bit identifier (unique)
- Key : k-bit identifier (unique)
- Value : bytes (can be a file, an IP, ...)

Generic DHT (Distributed Hash Table)

- put(key, value)
  - Stores (key, value) on the node responsible of *key*
- value = get(key)
  - Retrieves the data associated with *key*

Current implementations

Software

- Kadmelia
- Chord
- CAN

Usage

- File sharing
- Naming
- Chat service
- Databases

Still limited

Fundamental Problems

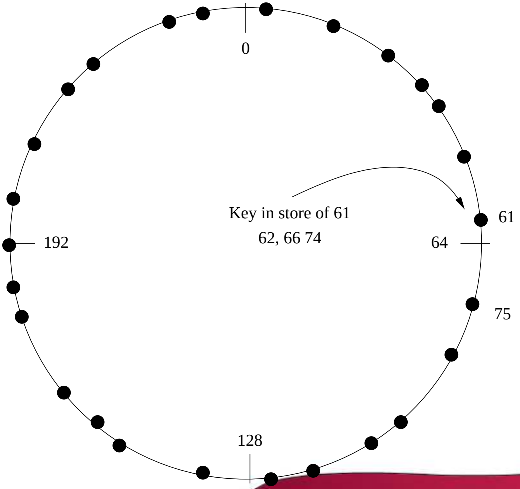
- Complex request
- Data coherence
- Request with several answer

Implementation difficulties

- Distribute workload evenly
  - Keys
  - Requests
- Only local information
- Dynamic information

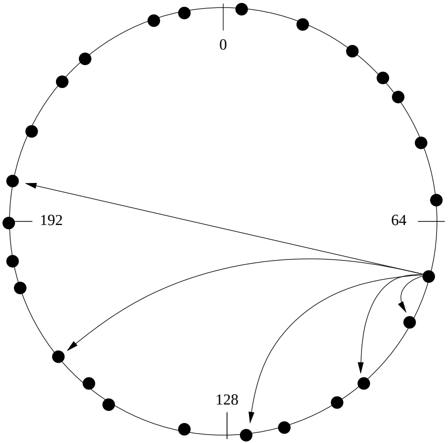
Chord structure

- Nodes are distributed on a circle
- Keys are assigned to the node with *Id* just before their value



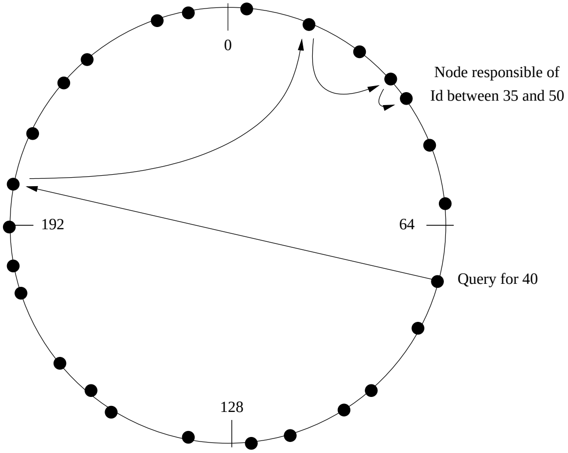
Neighbors

- $Log(N)$ neighbors
- Neighbors are nodes $Id + 1, Id + 2, Id + 4, ..., Id + 2^i, ..., Id + 2^{k-1}$ (modulo 2^k).



Routing algorithm

- Forward to the neighbor which is prior to the key
- Query needs at most $Log(N)$ messages



Chord characteristics

Efficient

- If a (key, value) exists, the query will find it
- Fast : $Log_2(1.000.000) = 23$
- Small neighbors table $Log_2(N)$

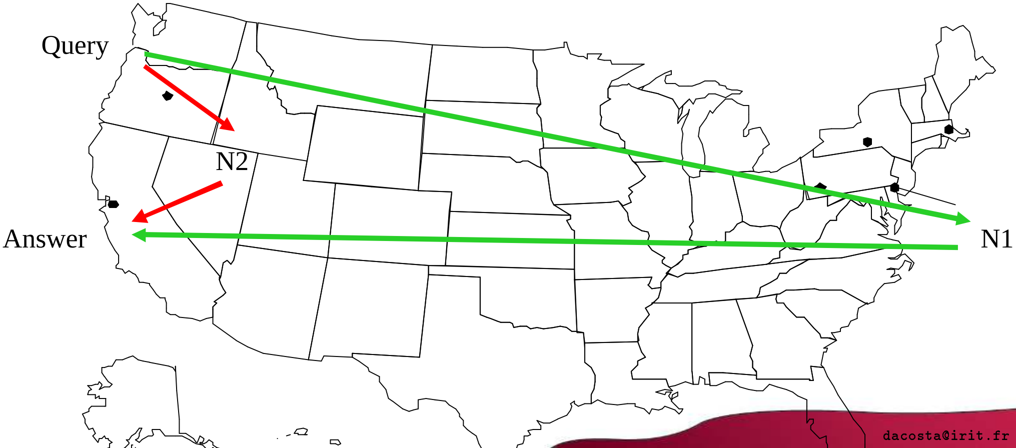
Chord characteristics

Some problems

- Security and privacy
- Attack
- How to test and evaluate such system ?
- Real performance (instead of number of messages)

Physical overlay

- Logical topology mapped in the physical network :



Plan

1. What is P2P
2. First generation systems
3. Self-organized systems
4. Structured systems
5. Distributed Hash Table
6. Conclusion

Conclusion

- Peer to Peer systems are efficient for several uses (using border resources)
- Recent systems are scalable
- Low cost alternative to Client/Server
- Field old enough to be used in real cases
- Still not perfect
  - Trust & certification
  - Anonymity
  - Security
  - Performance
  - Layers fees

When to use Peer to Peer systems

- Limited budget
- Large audience
- Trusted users
- Dynamic system, but not too much
- Do not need guarantee
- Do not need control

Vision of the future

User centered

No more servers
All content provided and served by users

- Only cooperation of peers
  - Wikipedia
  - Social networks
  - Youtube
  - *Good Ol' Time* web-pages