1

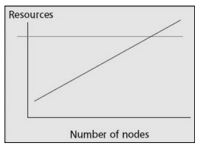
#02 Peer to Peer Networking

CLIENT/SERVER COMPUTING AND WEB TECHNOLOGIES

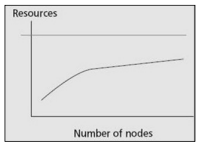
The architectures

2

- Server-based architecture
  - Client-Server / Server-Cluster
  - Problems：
    - Limited resources
    - All loads are centered on the server
  - Server-based architecture has low scalability.
  - The setup and maintenance cost is high.



- Peer-to-Peer (P2P) architecture
  - Advantages：
    - Distributing loads to all users
    - Users consume and provide resources
  - P2P architecture has high scalability.
  - The setup and maintenance cost is low.



Peer-to-peer (P2P)

3

"Peer-to-peer is a way of structuring distributed applications such that the **individual nodes have symmetric roles**. Rather than being divided into clients and servers each with quite distinct roles, in P2P applications **a node may act as both a client and a server**."

-- Charter of Peer-to-peer Research Group, IETF/IRTF, June 24, 2004 (http://www.irtf.org/charters/p2prg.html)

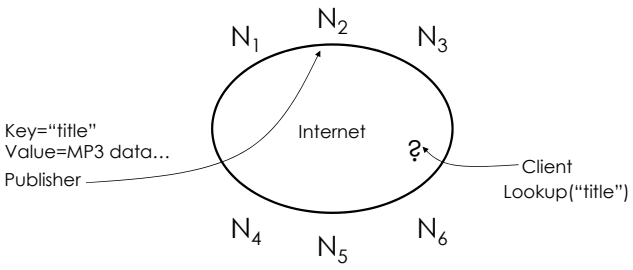
Classification of P2P systems

4

- Hybrid P2P – Preserves some of the traditional C/S architecture. A central server links between clients, stores indices tables, etc
  - **Napster**
- Unstructured P2P – no control over topology and file placement
  - **Gnutella, Morpheus, Kazaa, etc**
- Structured P2P – topology is tightly controlled and placement of files are not random
  - **Chord, CAN, Pastry, Tornado, etc**

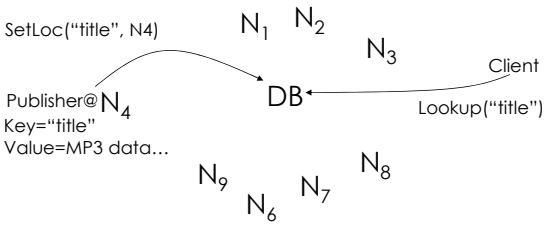
The lookup problem

5



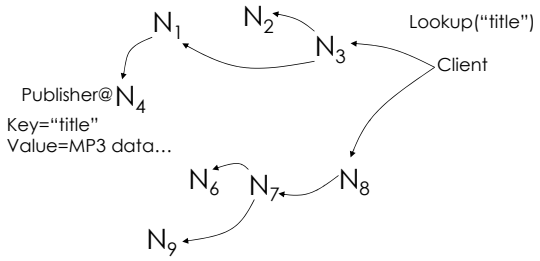
Centralized lookup (Napster)

6



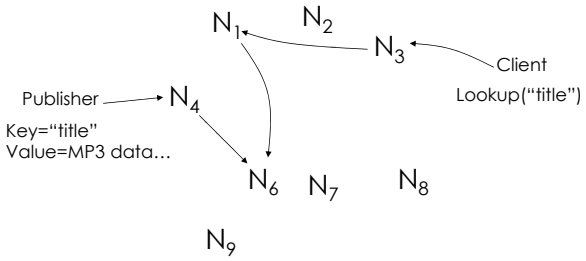
Simple, but O(*N*) state and a single point of failure

Flooded queries (Gnutella)



Robust, but worst case O(*N*) messages per lookup

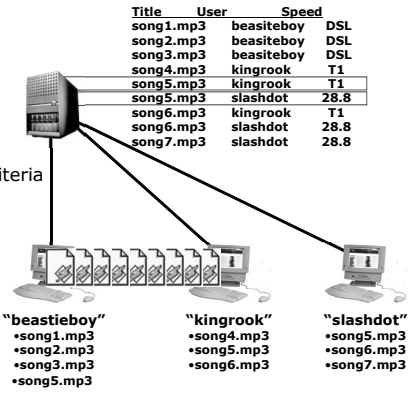
Routed queries (Freenet, Chord, etc.)



Napster Sharing Style:

hybrid center + edge

1. Users launch Napster and connect to Napster server
2. Napster creates dynamic directory from users' personal .mp3 libraries
3. ***beastieboy*** enters search criteria
4. Napster displays matches to ***beastieboy***
5. ***beastieboy*** makes direct connection to ***kingrook*** for file transfer



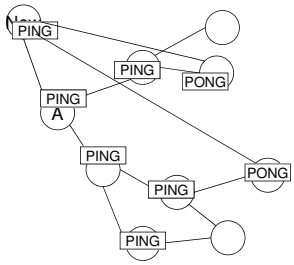
Title	User	Speed
song1.mp3	beasiteboy	DSL
song2.mp3	beasiteboy	DSL
song3.mp3	beasiteboy	DSL
song4.mp3	kingrook	T1
song5.mp3	kingrook	T1
song5.mp3	slashdot	28.8
song6.mp3	kingrook	T1
song6.mp3	slashdot	28.8
song7.mp3	slashdot	28.8

Gnutella Protocol

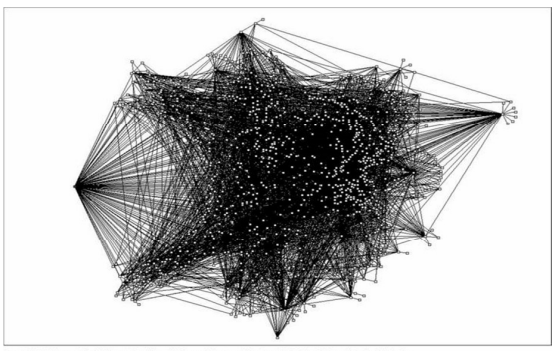
Scenario: Joining Gnutella Network

- The new node connects to a well known 'Anchor' node or 'Bootstrap' node.
- Then sends a PING message to discover other nodes.
- PONG messages are sent in reply from hosts offering new connections with the new node.
- Direct connections are then made to the newly discovered nodes.

Gnutella Network



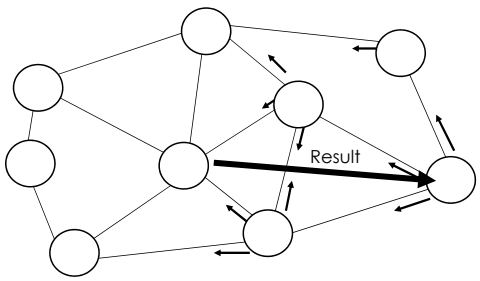
Topology of a Gnutella Network



Source: Mihajlo A. Jovanovic, Fred S. Annexstein, and Kenneth A. Berman, Laboratory of Networks and Applied Graph Theory, University of Cincinnati.

Gnutella:

Flood the Request



Fully distributed storage and directory!

So Far/We Want

So Far

- Centralized :
  - Directory size – O(n)
  - Number of hops – O(1)
- Flooded queries:
  - Directory size – O(1)
  - Number of hops – O(n)

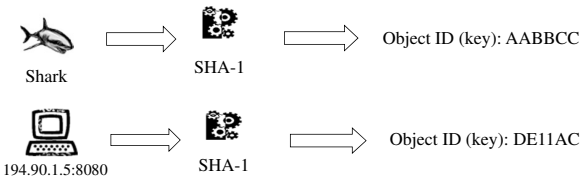
We Want

- Efficiency : O(log(n)) messages per lookup
- Scalability : O(log(n)) state per node
- Robustness : surviving massive failures

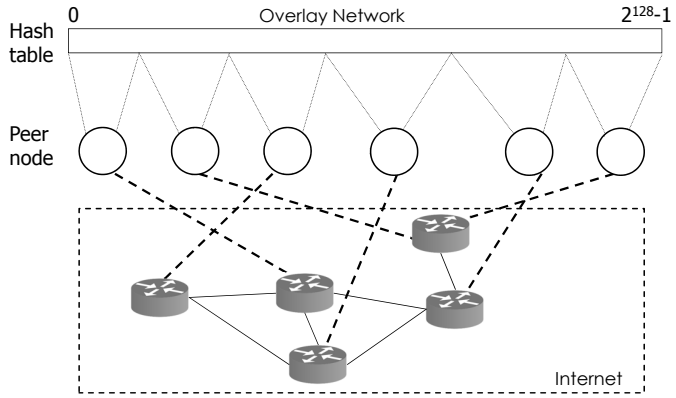
n: number of participating nodes

How Can It Be Done?

- How do you search in O(log(n)) time?
  - Binary Search
    - You need an ordered array
    - How can you order nodes in a network and data objects?
  - Hash Function



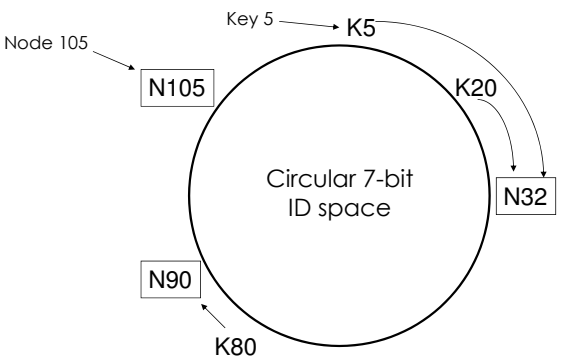
Viewed as a Distributed Hash Table



DHT

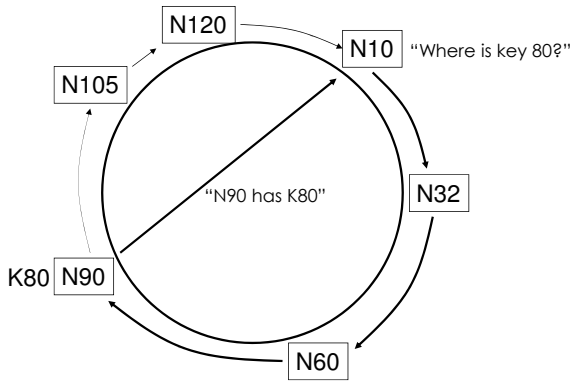
- Distributed Hash Table
- Input: key (file name)
  Output: value (file location)
- Each node is responsible for a range of the hash table, according to the node's hash key. Objects' directories are placed in (managed by) the node with the closest key
- It must be adaptive to dynamic node joining and leaving

Consistent hashing [Karger 97]

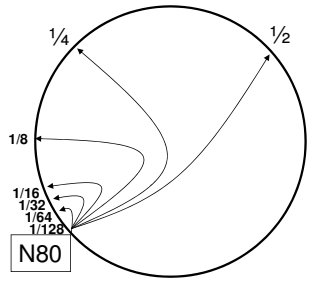


A key is stored at its successor: node with next higher ID

Basic lookup



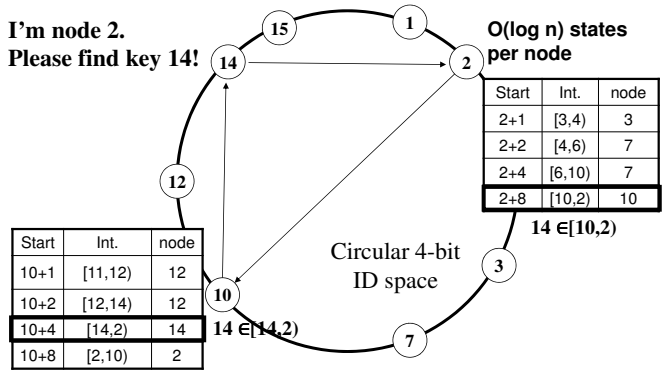
"Finger table" allows log(N)-time lookups



Chord Lookup

O(log n) hops (messages) for each lookup!!

I'm node 2. Please find key 14!



O(log n) states per node

Start	Int.	node
2+1	[3,4)	3
2+2	[4,6)	7
2+4	[6,10)	7
2+8	[10,2)	10

Start	Int.	node
10+1	[11,12)	12
10+2	[12,14)	12
10+4	[14,2)	14
10+8	[2,10)	2

P2P Content Distribution

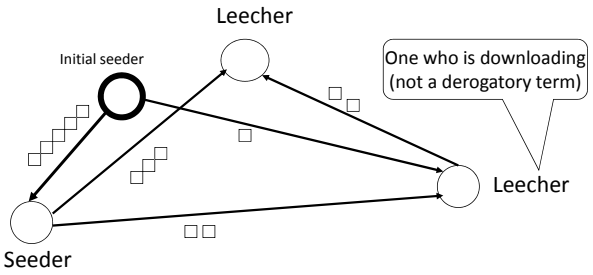
- BitTorrent builds a network for every file that is being distributed.
- Big advantage of BitTorrent:
  - Can send "link" to a friend
  - "Link" always refers to the same file
- Not really feasible on Napster, Gnutella, or KaZaA
  - These networks are based on searching, hard to identify a particular file
  - Downside of BitTorrent: No searching possible
    - Websites with "link collections" and search capabilities exist

BitTorrent

- Efficient content distribution system using **file swarming**. **Does not** perform all the functions of a typical p2p system, like searching.
  - A swarm is the set of peers that are participating in distributing the same files
- To share a file or group of files
  - the initiator first creates a *.torrent* file, a small file that contains
    - Metadata about the files to be shared, and
    - Information about the tracker, the computer that coordinates the file distribution.
  - Downloaders first obtain a *.torrent* file, and then connect to the specified tracker, which tells them from which other peers to download the pieces of the file.

BitTorrent Lingo

- Seeder = a peer that provides the complete file.
- Initial seeder = a peer that provides the initial copy.



References

- Robert Morris, Ion Stoica, David Karger, M. Frans Kaashoek, Hari Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications"
- J. R Jiang, "P2P Networking"
- Sukumar Ghosh, "The BitTorrent Protocol"