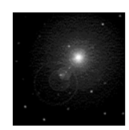
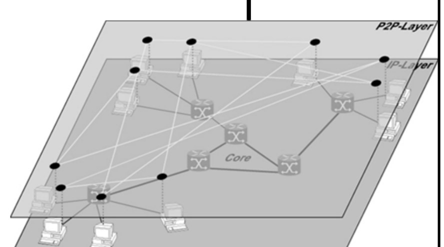
Next Class
Hybrid Gnutella 0.6



FOUNDATION OF PEER-TO-PEER SYSTEMS

Overview

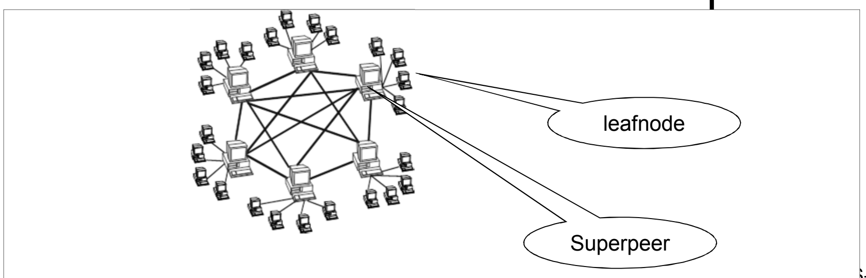
1. General Characteristics of Early Peer-to-Peer Systems
2. Centralized Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
3. Pure Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
4. Hybrid Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion



FP2P13F, javed@kent.edu

Definition of Hybrid P2P

- Main characteristic, compared to pure P2P: Introduction of another dynamic hierarchical layer
- Hub based network
- Reduces the signaling load without reducing the reliability
- Election process to select an assign Superpeers
- Superpeers: high degree (degree>>20, depending on network size)
- Leafnodes: connected to one or more Superpeers (degree<7)



FOUNDATION OF PEER-TO-PEER SYSTEMS

Model of Hybrid P2P Networks

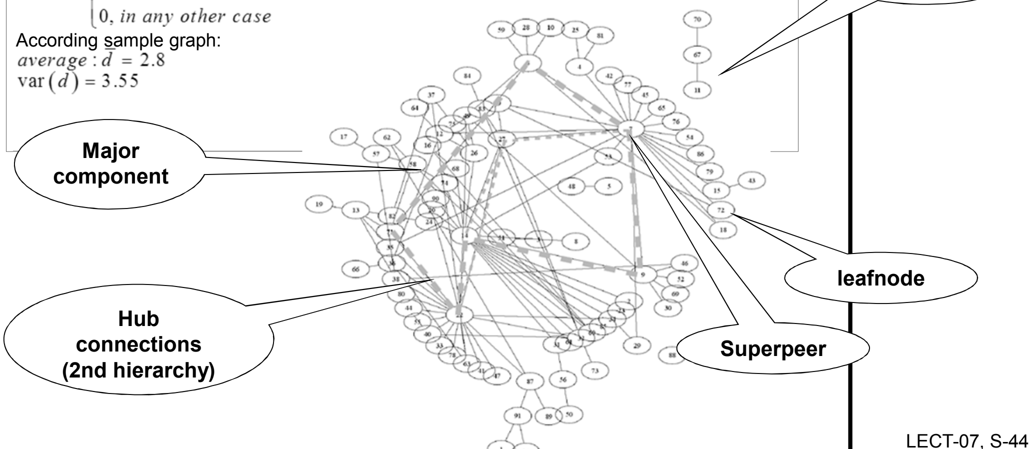
Degree distribution:

$$p(d) = \begin{cases} c \cdot d^{-1.4}, 1 < d \leq 7 \\ c \cdot 1^{-1.4} - 0.05, d = 1 \\ c \cdot 0.05, d = 20 \\ 0, \textit{in any other case} \end{cases}, \textit{with } c = \left(\sum_d \frac{p(d)}{c} \right)^{-1}$$

According sample graph:

$average : \bar{d} = 2.8$

$\mathrm{var}(\bar{d}) = 3.55$



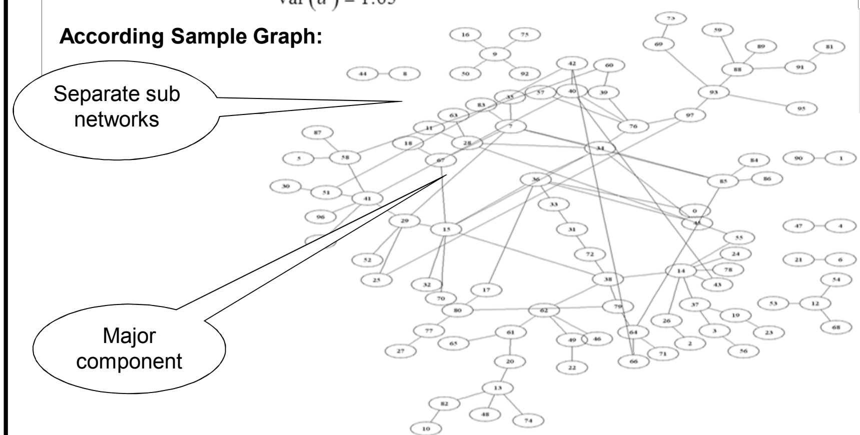
Flashback: Degree Distribution in Pure P2P Networks

Degree distribution:

$$p(d) = \begin{cases} c \cdot d^{-1.4}, \; 0 < d \le 7 \\ 0, \; in\ any\ other\ case \end{cases}, \; with\ c = \left(\sum_{d} \frac{p(d)}{c} \right)^{-1}$$

$$average: \bar{d} = 2.2$$
$$\mathrm{var}(d) = 1.63$$

According Sample Graph:



FOUNDATION OF PEER-TO-PEER SYSTEMS

Basic Characteristics of Hybrid P2P

- Bootstrapping:
  - Via bootstrap-server (host list from a web server)
  - Via peer-cache (from previous sessions)
  - Via well-known host
  - Registration of each leafnode at the Superpeer it connects to, i.e. it announces its shared files to the Superpeer
- Routing:
  - Partly decentralized
    - Leafnodes send request to a Superpeer
    - Superpeer distributes this request in the Superpeer layer
    - If a Superpeer has information about a matching file shared by one of its leafnodes, it sends this information back to the requesting leafnode (backward routing)
  - Hybrid protocol (reactive and proactive): routes to content providers are only established on demand; content announcements from leafnodes to their Superpeers
  - Requests: flooding (limited by TTL and GUID) in the Superpeer layer
  - Responses: routed (Backward routing with help of GUID)
- Signaling connections (stable, as long as neighbors do not change):
  - Based on TCP
  - Keep-alive
  - Content search
- Content transfer connections (temporary):
  - Based on HTTP
  - Out of band transmission (directly between leafnodes)

FOUNDATION OF PEER-TO-PEER SYSTEMS

Example: Gnutella 0.6

Example: Gnutella 0.6

- Program for sharing files over the Internet
- Focus:
  - decentralized method of searching for files
  - Higher signaling efficiency than Pure P2P
  - Same reliability (no single point of failure)
- Basis of most file-sharing applications (not BitTorrent)
- Brief History:
  - **Spring 2001**: resulted from Gnutella 0.4 by further developments to improve scalability → Gnutella 0.6 (Hybrid P2P)
  - Since then:
    - available in a lot of implementations (Limewire, bearshare,…)
    - Developed further on (privacy, scalability, performance,…)

Gnutella 0.6 Network Organization

New connection/network setup

- Upon connection to the network via a Superpeer, each node is a leafnode
- It announces its shared content to the Superpeer it connected to
- Superpeer thus updates its routing tables
- Election mechanism decides which node becomes a Superpeer or a leafnode (depending on capabilities (storage, processing power) network connection, the uptime of a node,…), if
    - Too many nodes are connected to one Superpeer
    - A Superpeer leaves the network
    - To less nodes are connected to a Superpeer

Concept: Ultra Peers

- It is a scheme to have a hierarchical Gnutella network by categorizing the nodes on the network as leaves and ultrapeers. An ultrapeer acts as a proxy to the Gnutella network for the leaves connected to it.

- This has an effect of making the Gnutella network scale, by reducing the number of nodes on the network involved in message handling and routing, as well as reducing the actual traffic among them.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Ultrapeer Election

- Since Gnutella is a decentralized system, ultrapeers are **elected** without the use of a central server. It is up to each node to determine if it is to become an ultrapeer or a shielded leaf node.
- Some Basic Requirements:
  - Not firewalled.
  - Sufficient downstream and upstream bandwidth.
  - Sufficient uptime
  - Sufficient RAM and CPU speed.
- If the above criterion are met, a node is said to be ultrapeer capable. When either an ultrapeer capable node will actually become an ultrapeer depends on if there is need for more ultrapeers on the network, and on how well the above criterion are met.

Ultrapeer Messages

- Ultrapeer capabilities and information are exchanged during the handshaking sequence when trying to establishing a new Gnutella connection. The following new headers are used in handshake:
- X-Ultrapeer: "True"
  - signals that node is an ultrapeer, "False" signals that the node wants to be a shielded leaf node.
- X-Ultrapeer-Needed:
  - Used to balance the number of ultrapeers.
- X-Try-Ultrapeers:
  - contains only addresses of ultrapeers.
- X-Query-Routing:
  - Signals support for the Query Routing Protocol

FOUNDATION OF PEER-TO-PEER SYSTEMS

FP2P13F, javed@kent.edu
Javed I. Khan@2008

Handshake Messages

- A leaf is trying to connect to a Ultrapeer.

Leaf	Ultrapeer
GNUTELLA CONNECT/0.6 User-Agent: LimeWire/1.0 X-Ultrapeer: False X-Query-Routing: 0.1	
	GNUTELLA/0.6 200 OK User-Agent: LimeWire/1.0 X-Ultrapeer: True [note error in RFC] X-Ultrapeer-Needed: False X-Query-Routing: 0.1 X-Try: 24.37.144:6346, 193.205.63.22:6346 X-Try-Ultrapeers: 23.35.1.7:6346, 18.207.63.25:6347
GNUTELLA/0.6 200 OK	

- The leaf is now a shielded node of the ultrapeer. The leaf should drop any non ultrapeer connections and send a QRP routing table (assuming QRP is used).

FOUNDATION OF PEER-TO-PEER SYSTEMS

Example Handshake Messages

- A leaf is trying to connect to another leaf.

New Leaf	Existing Leaf
GNUTELLA CONNECT/0.6 X-Ultrapeer: False	
	GNUTELLA/0.6 503 I am a leaf X-Ultrapeer: False X-Try: 24.37.144:6346 X-Try-Ultrapeers: 23.35.1.7:6346
	[DROP CONNECTION]

- If a shielded leaf node receives a connection request, it will refuse to accept the connection by returning a 503 error code together with X-Try and X-Try-Ultrapeer headers to redirect the remote host to other addresses.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Example Handshake Messages

- A leaf is trying to connect to another leaf.

New Leaf	Existing Leaf
GNUTELLA CONNECT/0.4 X-Ultrapeer: False	
	GNUTELLA/0.6 200 OK X-Ultrapeer: False
GNUTELLA/0.4 200 OK	

- Sometimes nodes will be ultrapeer-incapable but unable to find an ultrapeer. In this case, they behave exactly like old, unrouted Gnutella 0.4 connections.

Example Handshake Messages

- When two ultrapeers meet, both set X-Ultrapeer: true.

Ultrapeer A	Ultrapeer B
GNUTELLA CONNECT/0.6 X-Ultrapeer: True	
	GNUTELLA/0.6 200 OK X-Ultrapeer: True
GNUTELLA/0.6 200 OK	

- If both have leaf nodes, they will remain ultrapeers after the interaction. No QRP route table is sent between ultrapeers.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Example Handshake Messages

- Sometimes there will be too many ultrapeer-capable nodes on the network. Consider the case of an ultrapeer A connecting to an ultrapeer B.

Ultrapeer A	Ultrapeer B
GNUTELLA CONNECT/0.6 X-Ultrapeer: True	
	GNUTELLA/0.6 200 OK X-Ultrapeer: True X-Ultrapeer-Needed: False
GNUTELLA/0.6 200 OK X-Ultrapeer: False	

- If B doesn't have enough leaves, it may direct A to become a leaf node. If A has no leaf connections, it stops fetching new connections, drops any Gnutella 0.4 connections, and sends a QRP table to B. Then B will shield A from all traffic. If A has leaf connections, it ignores the guidance, as in the above case.

Query Routing Protocol (QRP)

- The Query Routing Protocol (QRP for short) is an essential part of the Ultrapeer specification: it governs how the Ultrapeer will filter queries and only forward those to the leaf nodes most likely to have a match.

- This is done without even knowing the resource names, by looking the query words through a big hash table, that is sent by the leaf node to its Ultrapeer.

- The aim of the QRP is to avoid forwarding a query that cannot match, it is not to forward only those queries that will match.

QRP Leaf Node Role

- Break all the resource names into individual words. A word is made of a consecutive sequence of letters and digits.
- Hash each word with a well-known hash function and insert a "present" flag in the corresponding hash table slot. Note that this hash table is a big array, and we don't store the key, only the fact that a key ended up filling some slot. All words are lower-cased and all accents are removed from them, i.e. "déjà" is transformed into "deja", so that only ASCII characters remain. Only those words that are made of at least 3 letters are retained.
- All words are re-hashed with their trailing 1, 2, or 3 letters removed, provided the word length after such trimming is at least 3 letter long. This is a simple attempt to remove plural from words. Optionally, nodes can chop off more letters from the end, provided that each hashed word is at least 3 character long.
-
- The "boolean vector" built at later stage is optionally compressed, broken up in small messages, and sent mixed with regular Gnet traffic to the ultrapeer.

FOUNDATION OF PEER-TO-PEER SYSTEMS

LECT-07, S-59
FP2P13F, javed@kent.edu
Javed I. Khan@2008

QRP Ultrapeer Role

- Until the whole "boolean vector" is received from a leaf node, all queries are forwarded to that node.
- When the "boolean vector" is fully received, it is going to be used as the Query Routing table for that leaf node: queries are broken into individual words, all accentuated letters are removed.
- For each leaf node with a Query Routing table:
  - Each word is then hashed and looked up in the Query Routing table.
  - Depending on the query matching rules either ALL the words will be required to be found in the Query Routing, or only some of them, to declare a Query Routing Hit.
  - Only those queries that were declared a Hit at the previous stage will be forwarded to a given leaf node.

FOUNDATION OF PEER-TO-PEER SYSTEMS

LECT-07, S-60
FP2P13F, javed@kent.edu
Javed I. Khan@2008

QRP Messages

- ROUTE_TABLE_UPDATE (0x30), Reset variant (0x0): to clear the routing table and to set a new routing table for one leafnode

Variant	Table_Length	Infinity

ield Name	Bytes	Meaning
VARIANT	1	The message variant. Always 0x0 for RESET.
TABLE_LENGTH	4	The length of the sender's route table, i.e., the number of entries. *(Earlier versions of this document incorrectly stated the meaning of this value.)* For hashing reasons, this *must be a power of 2.*
INFINITY	1	The route table value for infinity, i.e., the maximum distance to any file in the table+1.

FOUNDATION OF PEER-TO-PEER SYSTEMS

QRP Messages

- ROUTE_TABLE_UPDATE (0x30), Patch variant(0x1): to update and set a new routing table with a certain number of entries (e.g. new shared files)

0	1	2	3	4	5 n+4
Variant	Seq_No	Seq_Size	Compressor	Entry_Bits	DATA

Field Name	Bytes	Meaning
VARIANT	1	The message variant. Always 0x1 for PATCH.
SEQ_NO	1	The position of this message in the update sequence.
SEQ_SIZE	1	The total number of messages in this update sequence.
COMPRESSOR	1	The algorithm to use when decompressing data. Currently defined values: 0x0 no compression, 0x1 ZLIB compression
ENTRY_BITS	1	The number of bits per uncompressed patch entry, including the sign bit. Must be 4 or 8.
DATA	to end	The compressed table patch.

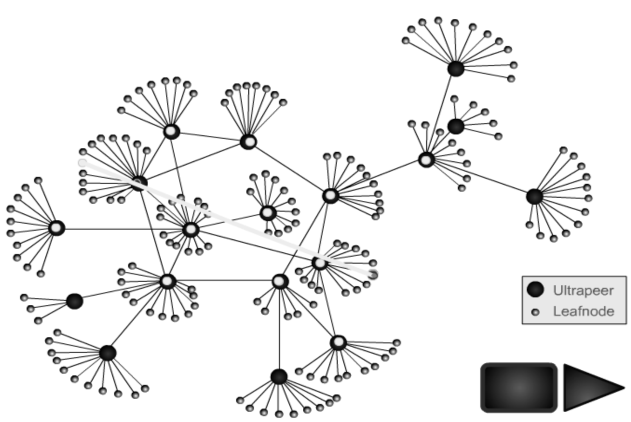
FOUNDATION OF PEER-TO-PEER SYSTEMS

Gnutella 0.6 Routing

- Content requests:
  - Leafnode sends request to Superpeer
  - Superpeer looks up in its routing tables whether content is offered by one of its leafnode. In this case the request is forwarded to this node.
  - Additionally the Superpeer increases the hopcounter and forwards this request to the Superpeers it is connected to.
  - To enable backward routing, the peer has to store the GUID of the message connected to the information from which peer it received the request in the previous hop
  - If a Superpeer receives such a request from another Superpeer, this request is handled the same way, as if it would have received it from one of its leafnodes
  - After the hopcounter of the request reaches the TTL-value it is not forwarded any further (prevent circles)
- Content responses:
  - If a leafnode receives a request, it double-checks whether it shares the file (should be the case, as long as the routing tables of the Superpeer are correct)
  - In case of success, the leafnode sends a content reply back to the requesting peer, by sending it back to that node (Superpeer) it received the message from (backward routing)
  - Hop by hop the message can thus be routed back to the requesting node
- Content exchange:
  - Directly between the leafnodes, via HTTP connections

FOUNDATION OF PEER-TO-PEER SYSTEMS

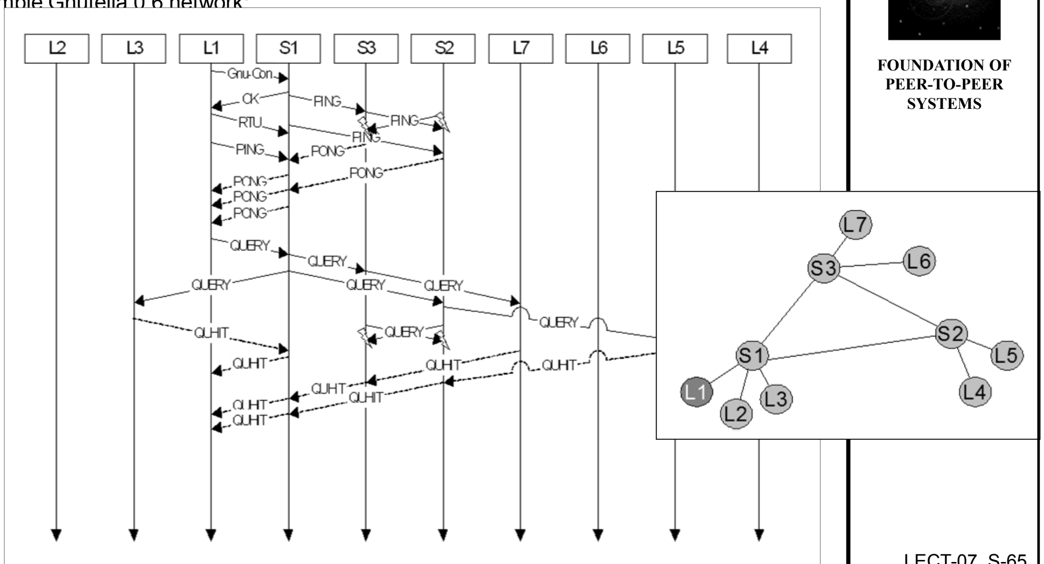
Gnutella 0.6: How Does It Work



FOUNDATION OF PEER-TO-PEER SYSTEMS

Summary of the Signaling in Gnutella 0.6

Sample Gnutella 0.6 network:

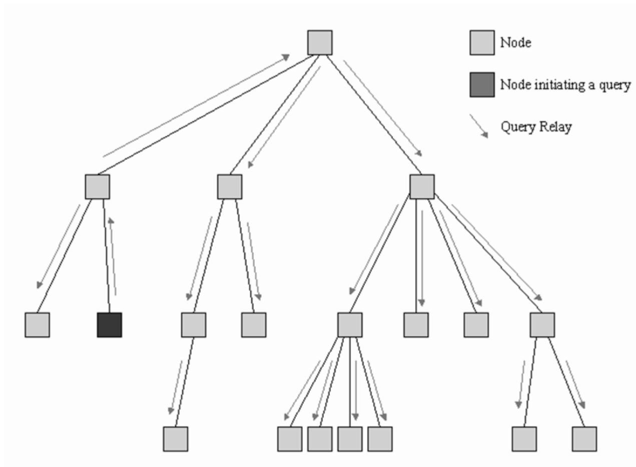


FOUNDATION OF PEER-TO-PEER SYSTEMS

FP2P13F, javed@kent.edu

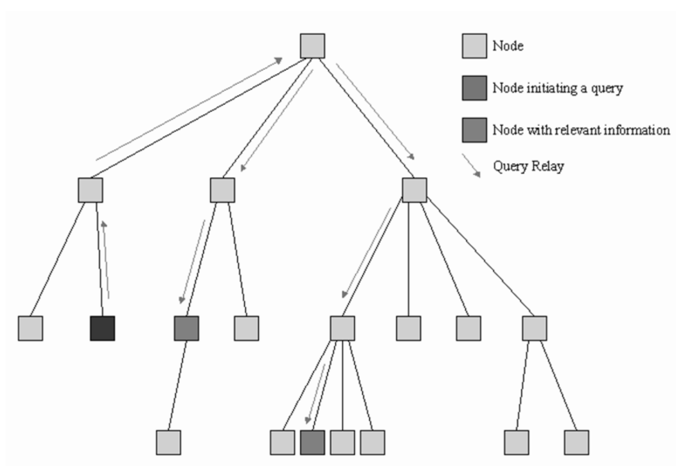
Prinkey- Rhor's Content-based Query Routing

Basic Flood Query Routing



FOUNDATION OF PEER-TO-PEER SYSTEMS

Optimum Routing



FOUNDATION OF PEER-TO-PEER SYSTEMS

Condition for Optimum Routing

- For optimum routing all parents should precisely know the content of all its children.
- This will however, require huge amount of information to be sent and shared defeating the very purpose.
- A hash based scheme which generates no false negative, but may have false positive can still be used to scale.

FOUNDATION OF PEER-TO-PEER SYSTEMS

A Simple Hash Mask

- We need to choose a large enough bitmask to result in a small collision rate.
- A 2^{32} (about 4 billion) bit index space would give very low probability of collisions (e.g., there are only about 300,000 words in the English). However, this bitmask would then be over 500 MBs in size!
- For pragmatic reasons, the index space is limited to 2^{18} to 2^{20} entries which leads to 32 KB to 128 KB bitmasks, respectively.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Hash Mask Aggregation and Propagation

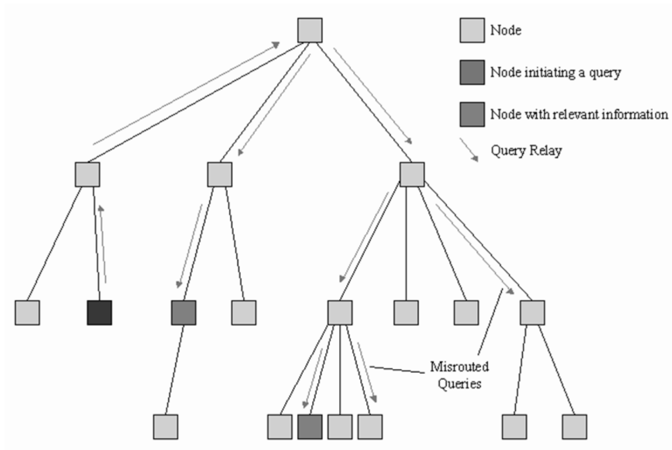
- Each node passed on its bitmask to its parent node.
- The parent node remembers each of the bitmasks from its connections.
- Parent also takes its own bitmask and logically ORs it with all bitmasks from children nodes. This aggregate bitmask is then passed up to its parent.
- The approach can be applied recursively up the tree. The bitmask size stays same.
- The update is propagated periodically by some keep-alive scheme.

Routing with Hash Masks

- The parent tries to find all the keywords in the bit masks of its children.
- If a child node does not have hit for all of keywords in an incoming query, it is not routed there.
- Any query is routed towards the connection where there is a match for all bits in the query.

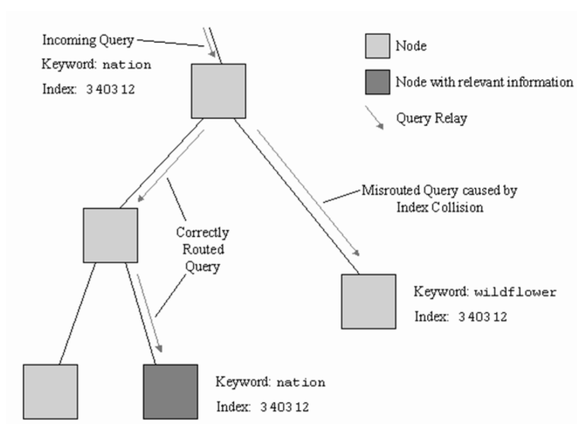
FOUNDATION OF PEER-TO-PEER SYSTEMS

Hash-Mask based Query



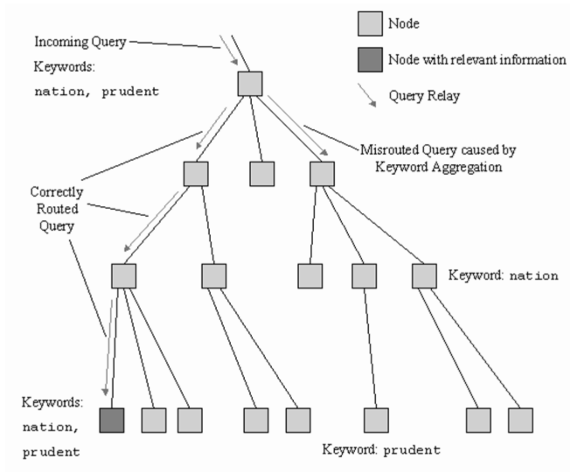
FOUNDATION OF PEER-TO-PEER SYSTEMS

Misrouting: Hash Collision



FOUNDATION OF PEER-TO-PEER SYSTEMS

Misrouting: Hash Aggregation



FOUNDATION OF PEER-TO-PEER SYSTEMS

How to Handle Old Nodes?

- There may be old (version) clients, which may not understand the hash-index scheme. Thus all query need to be propagated to them.
- Logically, the default bitmask for every old node will be all 1's. But it should not be ORed propagated.
- This also means that every uplinked bitmask will also be full. This will be a potential barrier to acceptance of the new protocol as a single "old" client connecting to a host will fill the bitmask for the entire branch and essentially ruin CQR.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Issues with Tree and Rhor’s Solution

- Prinkley’s basic scheme requires a tree topology with a designated root. This is difficult to form in a distributed network.
- Secondly, the node at the top of the tree to handle a disproportionate amount of traffic..
- In a non-tree network can we simply propagate route tables along all connections?
  - After T time steps, all hosts within T hops of a host X are aware of X’s files.- that’s OK.
  - But, the problem is that after T- X’s routing information will continue to propagate. This will dramatically increase the false positive rate- making in ineffective.

Hop Counting Hash

- For this reason, it is critical to limit the span of a host’s routing information.
- This can be done by associating a hops value with each keyword.
- Routing tables are now an array of numbers instead of an array of bits, where each number is the minimum distance to a matching file.

Counting Hash (QRT) Propagation

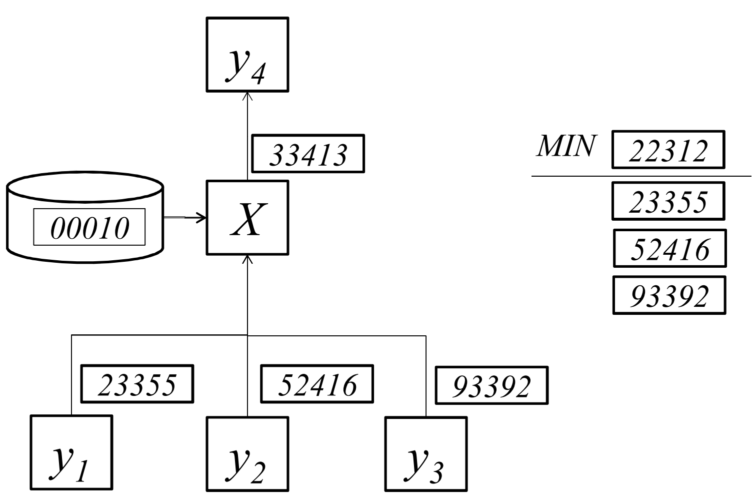
FOUNDATION OF PEER-TO-PEER SYSTEMS

- Let host X have connections to hosts $Y_1 \ldots Y_M$. Let $RT_{Yi\text{->}X}$ be the route table received by X from host Y_i.
- If X is not sharing any files with keywords hashing to h, the table $RT_{X\text{->}Yi}$ propagated by X to each host Y_i is given by $RT_{X\text{->}Yi}[h]=\min_{j \neq i}(RT_{Yj\text{->}X}[h])+1$.
- Otherwise (X itself has a file with keyword hashing to h), $RT_{X\text{->}Yi}[h]=1$.

LECT-07, S-79
FP2P13F, javed@kent.edu
Javed I. Khan@2008

Example QRT Propagation

FOUNDATION OF PEER-TO-PEER SYSTEMS



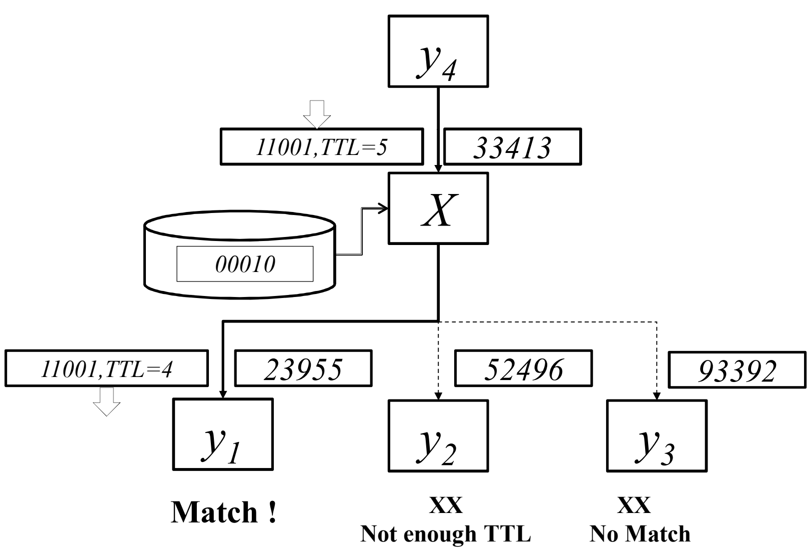
LECT-07, S-80
FP2P13F, javed@kent.edu
Javed I. Khan@2008

Rule for Query Propagation

- Let, for any connection C, $RT_C[i]$ is the number of hops to a file with a keyword that hashes to i, or infinity [9 in our example] if no such file exists.
- Note that $RT_C[i] \geq 1$, for all i.
- Queries are forwarded to those connections whose route tables have entries for all query keywords. That is, a query with TTL N and keywords K_1, .., K_M is only sent along a connection C if $RT_C[HASH(K_i)] \leq N$, for *all* $1 < i \leq M$.

FOUNDATION OF PEER-TO-PEER SYSTEMS

Example



FOUNDATION OF PEER-TO-PEER SYSTEMS

Example: [need to create]

- The network need not have a root and may have cycles.
- After one time step, hosts have exchanged routing tables for files one hops away.
- This is illustrated below on the left-hand side. Here sets of keywords are shown instead of arrays of hashed values for simplicity. For example, the table "{bad/1, joke/2}" would really be represented as the array [∞, 2, ∞, 1, ∞, ...] if "bad" hashed to 3 and "joke" hashed to 1. After a second time step, hosts have exchanged routing tables for files two hops away. This is illustrated below on the right-hand side. Note that A now has routing entries for all files. On the other hand, B is aware that no files are reachable through A.

Protocol Issues

- Instead of one bit per keyword, hosts exchange a lg(M) bit number, where M is the maximum TTL. So with a typical TTL of 10, neighbors exchange 4 times more data than in Prinkey's original scheme.
- Hosts should not send out more than one message per connection per T minutes. If a neighbor sends multiple message within a T-minute window, consolidate into a single message.
- If the route table RT to be sent to a host is very similar to the last table RT' sent, it may be advantageous to send an incremental update.

FOUNDATION OF PEER-TO-PEER SYSTEMS

New Gnutella Message

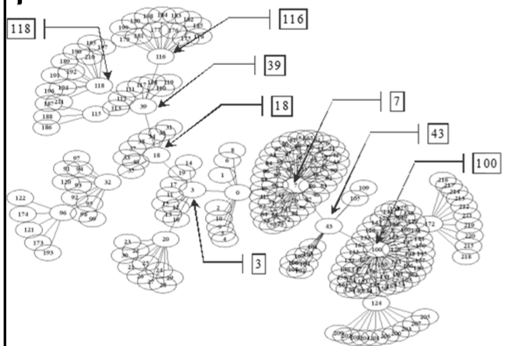
- To implement the above scheme for the Gnutella network suggests the new messages ROUTE_TABLE_UPDATE (code 0x20) with two variants (indicted in the first byte of payload)- RESET (0x0) and PATCH (0x1).

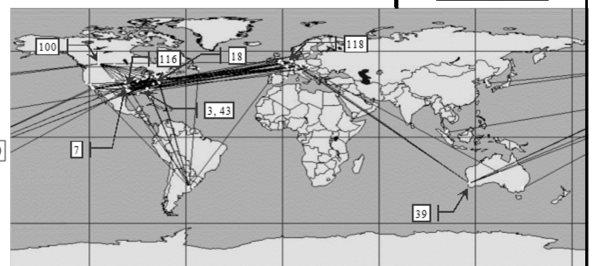
- RESET to set hop count for all keywords to infinity.

- PATCH to send incremental updates.

Advanced Topic:
Overlay Inefficiency

Topology of Hybrid P2P





Geographical view of a part of the Gnutella network (222 nodes); The numbers depict the node numbers from the abstract view (Figure on the left, measured on 01.08.2002)

Abstract network structure of a part of the Gnutella network (222 nodes Geographical view given by Figure on the right, measured on 01.08.2002

- Virtual network not matched to physical network. See path from node 118 to node 18.
- Superpeer (hub) structure clearly visible in abstract view

Discussion

- Disadvantages
  - Still High signaling traffic, because of decentralization
  - No definitive statement possible if content is not available or not found [dealing with incomplete information, seti, Asrar's work- javed]
  - Modem nodes may become bottlenecks
  - Overlay topology not optimal, as
    - no complete view available,
    - no coordinator
  - If not adapted to physical structure delay and total network load increases
    - Zigzag routes
    - Loops
  - Can not be adapted to physical network completely because of hub structure
  - Asymmetric load (Superpeers have to bear a significantly higher load)
- Advantages
  - No single point of failure
  - Can provide anonymity
  - Can be adapted to special interest groups
- Application areas [p2p techniques are becoming a layer than application-javed]
  - File-sharing
  - Context based routing (see chapter about mobility)

Further P2P systems based on hybrid P2P

- Edonkey
- Kazaa/FastTrack
- Emule
- OpenNap
- …

Summary

- P2P technologies offer an innovative overlay infrastructure for decentralized and distributed systems
- Due to the distributed nature, the signaling load is very high, but it can be reduced with introduction of hierarchies, compression and geo-sensitive protocols
- Advantages:
  - Simple basic principle
  - Enhanced reliability
  - Redundancy (high replication rate)
  - Unsusceptible against Denial of Service attacks (DOS)
  - No single point of failure
  - No central instances/administration
  - Direct and instantaneous communication possible
  - Large variety of applications possible