Applicazioni P2P

Corso di ***Applicazioni Telematiche***

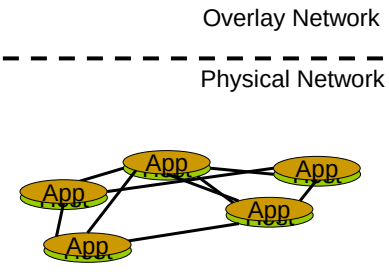
A.A. 2005-06 – Lezione n.17

Prof. Roberto Canonico

Università degli Studi di Napoli Federico II
Facoltà di Ingegneria

P2P Networks

- Use application layer overlay networks to carry protocol messages
- Nodes have equal functionality (i.e. share and search for resources)
- Evolution:
  from Unstructured to Structured



Caratteristiche di un sistema p2p

- Sistema distribuito nel quale ogni nodo ha identiche capacità e responsabilità e tutte le comunicazioni sono potenzialmente simmetriche;
- Peer to peer (obiettivi): condividere risorse e servizi (dove per risorse e servizi intendiamo: scambio di informazioni, cicli di CPU, spazio sul disco …);
- Scalabilità: il lavoro richiesto a un determinato nodo nel sistema non deve crescere (o almeno cresce lentamente) in funzione del numero di nodi nel sistema;
- Per migliorare la scalabilità sono nati i cosiddetti protocolli P2P di seconda generazione che supportano DHT (Distributed Hash Table);

Why is P2P Interesting

- Decentralised
  - Control
  - Resources (files, CPU)
- Robust
  - Auto reconfiguration of overlay network when nodes fail
  - No central failure point (i.e. cannot be switched off)
- Present new problems
  - Search for resources (files, services etc…)
  - Impact of P2P traffic on the Internet

Sistemi P2P: storia

- Proposti già da oltre 30 anni;
- Sviluppati nell'ultimo decennio;
- L'interesse verso questo tipo di protocolli è aumentato con la nascita dei primi sistemi per file-sharing (Napster (1999), Gnutella(2000));
- Nel 2000, 50 milioni di utenti hanno scaricato il client di Napster;
- Napster ha avuto un picco di traffico di circa 7 TB in un giorno;
- L'11/12/2002 è stata aperta l'asta online per la vendita del server di Napster;

Peer to Peer: What's the problem?

• Problem: how do we organize peers within ad-hoc, multi-hop pervasive P2P networks?

  • network of self-organizing peers organized in a decentralized fashion
  • such networks can rapidly expand from a few hundred peers to several thousand or even millions

• P2P Environment Recap:
  • Unreliable Environments
  • Peers connecting/disconnecting – network failures to participation
  • Random Failures e.g. power outages, Cable, DSL failure, hackers
  • Personal machines are much more vulnerable than servers
  • algorithms have to cope with this continuous restructuring of the network core.

• P2P systems need to treat failures as normal occurrences not freak exceptions
  • must be designed in a way that promotes redundancy with the tradeoff of a degradation of performance

Performance Issues in P2P Networks

3 main factors that make P2P networks more sensitive to performance issues:

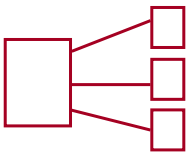
1. Communication.
   - Fundamental necessity
   - Users connected via different connections speeds
   - Multi-hop
2. Searching
   - No central Control so more effort is needed
   - Each hop adds to total bandwidth – problems: time outs
3. Equal Peers
   - Free Riders – unbalance in the harmonicity of network
   - Degrades performance for others
   - Need to get this right to adjust accordingly

Organizzazione dei peer: Topologie

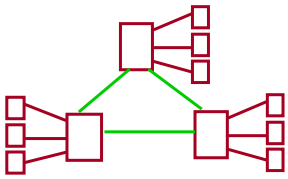
- Core
  - Centralized
  - Ring
  - Hierarchical
  - Decentralized
- Hybrid
  - Centralized-Ring
  - Centralized-Centralized
  - Centralized-Decentralized

Classificazione sistemi p2p: topologie

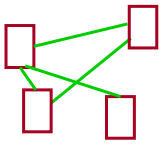
- Hybrid
  - Centralized index, P2P file storage and transfer



- Super-peer
  - A “pure” network of “hybrid” clusters

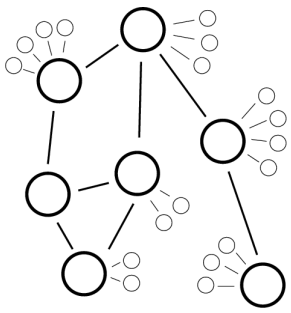


- Pure
  - functionality completely distributed



Centralized + Decentralized

- New Wave of P2P
- Clip2 Gnutella Reflector (next)
- FastTrack
  - KaZaA
  - Morpheus
- Email
- Like Social Networks perhaps ?



Sistemi P2P: fasi

- Nel funzionamento di una applicazione P2P di solito si possono individuare tre fasi principali:
  - Boot: permette a un peer di trovare la rete e di connettersi ad essa;
    - nessuno o quasi fa boot P2P
  - Lookup: permette ad un peer di trovare il gestore/responsabile di una determinata informazione;
    - pochi sono P2P, alcuni usano SuperPeer
  - Scambio di file;
    - sono tutti P2P, almeno in questo… ☺

Classificazione sistemi P2P: fasi

- Parleremo di applicazioni:
  - P2P pure se:
    - le fasi di boot, lookup e scambio di file sono P2P;
  - P2P se:
    - le fasi di lookup e scambio di file sono P2P;
    - la fase di boot utilizza qualche SERVER;
  - P2P Ibride se:
    - la fase di scambio dei file è P2P;
    - la fase di boot utilizza qualche SERVER;
    - nella fase di lookup vengono usati Peer particolari:

Hub (Direct Connect)	SuperPeer , Ultra Peer(Gnutella2)
Supernodo (KaZaA)	NodoRandezVous (JXTA)
MainPeer (EDonkey)	Server (WinMX)

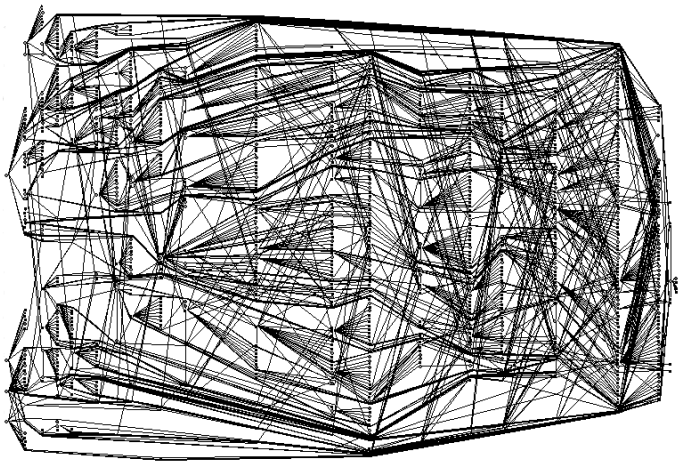
Why Look at Gnutella

- Widespread unstructured P2P network
  - Currently between 200,000 & 300,000 hosts
  - Popular Gnutella clients
    - LimeWire
    - Morpheus
    - BearShare
- Ideal as a research test bed
  - Large scale network demonstrates the need for scalable P2P protocols

Partial Map of Gnutella Network - 7/27/00

Clip2 Distributed Search Services

http://dss.clip2.com



Gnutella: caratteristiche generali

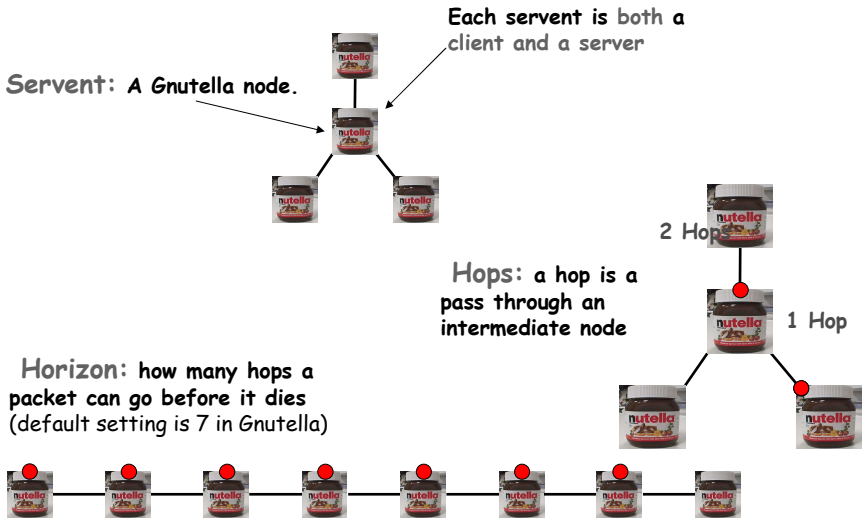
• Gnutella è un protocollo P2P;

• La lista degli host presenti in rete è disponibile sul server gnutellahost.com;

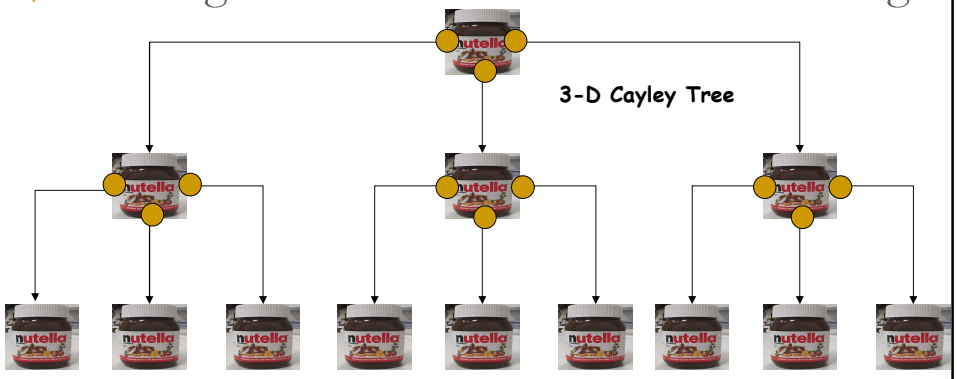
• Il Server gnutellahost.com(127.186.112.97) viene usato dai nodi per il boot:
  • Single point of failure;
  • Gnutella non è P2P Puro!!!;

• La Ricerca di un file usa il flooding:
  • controllo dei cicli;
  • TTL per evitare di congestionare la rete;

Gnutella jargon



Searching a Gnutella Network: Broadcasting



Searching in Gnutella involves broadcasting a Query message to all connected peers. Each connected peer will send it to their connected peers (say 3) and so on. Typically, this search will run 7 hops. If the number of connected peers, c=3 and the hops i.e. TTL=7 then the total number of peers searched (in a fully connected network) will be:

S = c + c2 +c3 + ….. ch = 3 + 9 + 27 + 81 + 243 + 729 + 2187 = 3279 Nodes

Gnutella Descriptors

- Gnutella messages that are passed around the Gnutella network

- **Ping**: used to actively discover hosts on the network.
  - A *servent* receiving a *Ping* descriptor is expected to respond with one or more *Pong* descriptors.
- **Pong**: the response to a *Ping*.
  - Each Pong packet contains a Globally Unique Identifier (GUID) plus address of *servent* and information regarding the amount of data it is making available to the network
- **Query**: the primary mechanism for searching the distributed network.
  - A *servent* receiving a *Query* descriptor will respond with a *QueryHit* if a match is found against its local data set.
- **QueryHit**: the response to a *Query*: contains IP address, GUID and search results
- **Push**: allows downloading from *firewalled servents*

Gnutella scenario

Step 0: Join the network

Step 1: Determining who is on the network

- "Ping" packet is used to announce your presence on the network.
- Other peers respond with a "Pong" packet.
- Also forwards your Ping to other connected peers
- A Pong packet also contains:
  - an IP address
  - port number
  - amount of data that peers is sharing
  - Pong packets come back via same route

Step 2: Searching

- Gnutella is a protocol for distributed search.
- Gnutella "Query" ask other peers if they have the file you desire (and have an acceptably fast network connection).
- A Query packet might ask, "Do you have any content that matches the string 'Homer"?
- Peers check to see if they have matches & respond (if they have any matches) & send packet to connected peers
- Continues for TTL

Step 3: Downloading

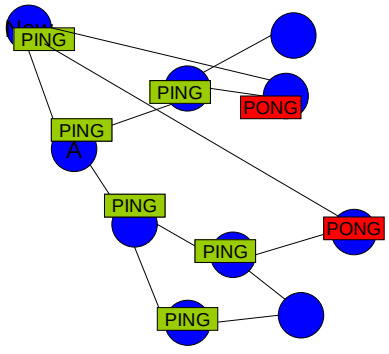
- Peers respond with a "QueryHit" (contains contact info)
- File transfers use direct connection using HTTP protocol's GET method
- When there is a firewall a "Push" packet is used – reroutes via Push path

Gnutella Protocol

Scenario: Joining Gnutella Network

- The new node connects to a well known 'Anchor' 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

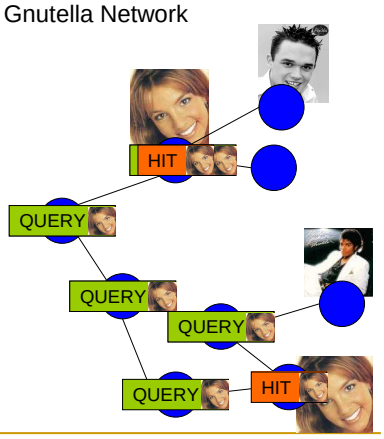


Gnutella Protocol

Scenario: Searching for a File

- A node broadcasts its QUERY to all its peers who in turn broadcast to their peers.
- Nodes route QUERYHITs along the QUERY path back to the sender containing file location details.
- To download files a direct connection is made using details of the host in the QUERYHIT messages.

Gnutella Network



Discovering Peers

• In the early days, used 'out of bounds' methods:

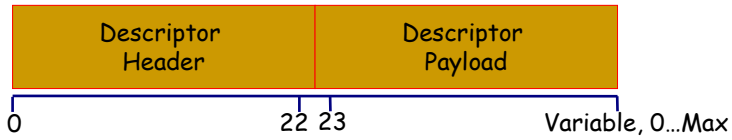
  • IRC (Internet Relay Chat) and asked users for hosts to connect to
  • Web pages – users checked a handful of web pages to see what hosts were available.

Users typed hosts into the Gnutella software until one worked…….

• Host Caches: e.g. *GnuCache* was used to cache Gnutella hosts and was included in *Gnut* software for unix

• Dynamically: by watching PING and PONG messages noting the addresses of peers initiating queries.

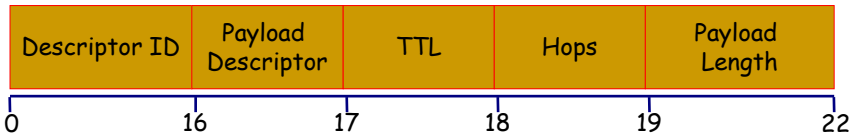
Gnutella Descriptors



Descriptor Types

- **Ping:** to actively discover hosts on the network.
- **Pong:** the response to a *Ping* (includes the GUID address of a connected *servent* and information regarding the amount of data it is making available to the network)
- **Query:** search mechanism
- **QueryHit:** the response to a *Query* (containing GUID and file info)
- **Push:** mechanism for *firewalled servents*

Gnutella Descriptor Header



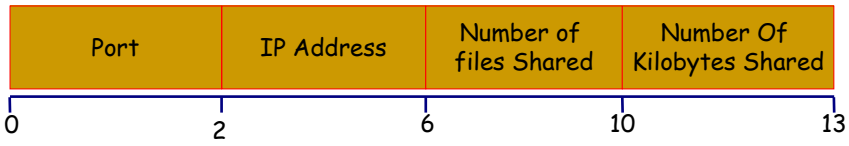
- **Descriptor ID:** a unique identifier for the descriptor on the network (16-byte string)
- **Payload Descriptor:** *0x00 = Ping: 0x01 = Pong: 0x40 = Push: 0x80 = Query: 0x81 = QueryHit*
- **TTL:** *Time To Live or Horizon.* Each *servent* decrements the TTL before passing it on - when TTL = 0, it is no longer forwarded.
- **Hops:** counts the number of hops the descriptor has traveled i.e. hops = TTL(0) when TTL expires

Payload Length: next descriptor header is located exactly *Payload Length* bytes from end descriptor header

Gnutella Payload 1 – Ping Descriptor

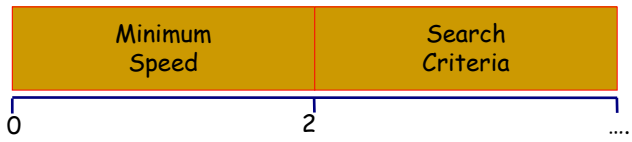
- **Ping descriptors:**
  - no associated payload
  - = zero length
- A Ping is simply represented by a ***Descriptor Header*** whose:
  - **Payload_ Length** field is 0x00000000.
  - ***Payload_Descriptor*** field = 0x00

Gnutella Payload 2 - Pong



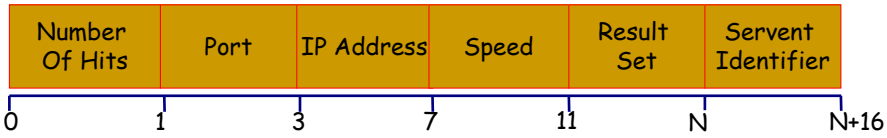
- **Port:** port which responding host can accept *incoming* connections.
- **IP Address:** IP address of the responding host (big-endian)
- **Number of Files Shared**: number of files responding host is sharing on the network
- **Number of Kilobytes Shared**: kilobytes of data responding host is sharing on the network.

Gnutella Payload 3 - Query



- **Minimum Speed:** minimum speed (in kb/second) of *servents* that should respond to this message.
  - A Servent receiving a Query descriptor with a minimum speed field of n kb/s should only respond with a QueryHit if it is able to communicate at a speed >= n kb/s
- **Search Criteria:** A nul (i.e. 0x00) terminated search string - maximum length is bound by *Payload_Length* field of the descriptor header.
  - e.g. "myFavouriteSong.mp3"

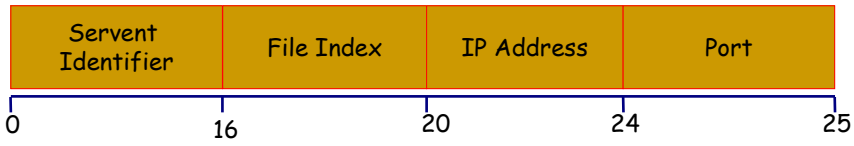
Gnutella Payload 4 - QueryHit



- **Number of Hits:** number of query hits in the result set
- **Port:** port which the responding host can accept incoming connections
- **IP Address:** IP address of the responding host (big-endian)
- **Speed:** speed (in kb/second) of the responding host
- **Result Set:** set of *Number_of_Hits* responses to the corresponding Query with the following structure:

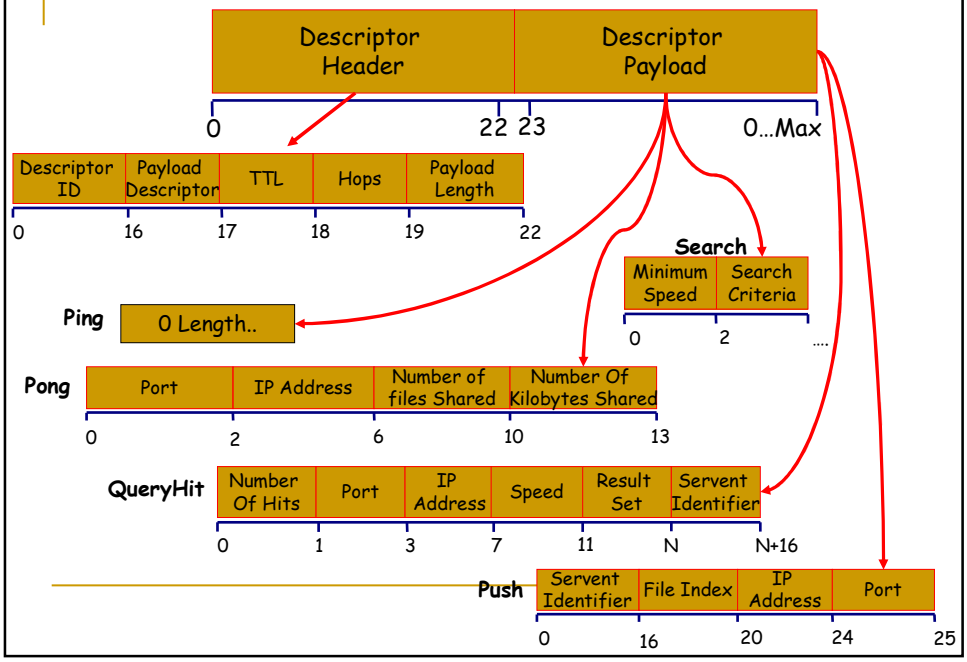
  - **File Index:** ID of file matching the corresponding query - assigned by the responding host
  - **File Size:** size (bytes) of this file
  - **File Name:** name of the file (double-nul (i.e. 0x0000) terminated)
- **Servent Identifier:** servent network ID (16-byte string), typically function of servent's network address - instrumental in the operation of the ***Push Descriptor*** ....

Gnutella Payload 5 - Push



- **Servent Identifier:** target *servent* network ID (16-byte string) requested to push file (with given index *File_Index*)
- **File Index:** ID of the file to be pushed from the target *servent*
- **IP Address:** IP address of target host which file should be pushed (big-endian forma)
- **Port:** port on target host which file should be pushed

Gnutella Descriptor



Problems With Gnutella

- Protocol scalability
  - Message broadcast technique imposes limitations on the network size

    packets per message = $\sum_{i=0}^{TTL} noPeers^{i}$
  - In November 2000 dial-up bandwidth barrier reached
- Overlay network efficiency
  - Random selection of peers results in inefficient use of the underlying network
  - Redundant traffic generated on the Internet

Current Client Optimisations

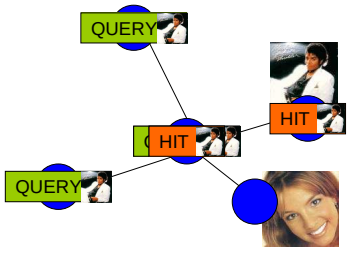
- PONG Caching
  - Eliminates frequent broadcasting of PING messages by reusing old PONG replies
- Hierarchical Overlay Structuring
  - Nodes join the network through gateways who filter PONG messages so the new node only connects with similar capacity nodes

Related P2P Research

- Unstructured P2P search techniques
  - Query Caching
  - Expanding Ring
  - Query Routing
  - Random Walks
- Overlay network construction
  - Clustering

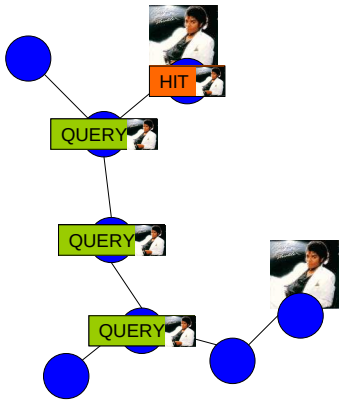
Query Caching

- Technique
  - Nodes may chose to respond to a QUERY message with someone else's QUERYHIT message that was seen in the past.
- Advantages
  - Reduces QUERY traffic for popular searches
- Disadvantages
  - May limit search scope



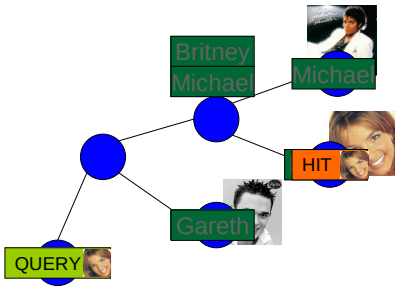
Expanding Ring

- Technique
  - The QUERY TTL is initially set low and increased for resending if no results are returned after a timeout period
- Advantages
  - Overall reduction in broadcast traffic
  - Automatically finds the max TTL
- Disadvantages
  - Longer delay for far away resources
  - More traffic generated in worst case where resources are far away (not characteristic of Gnutella)



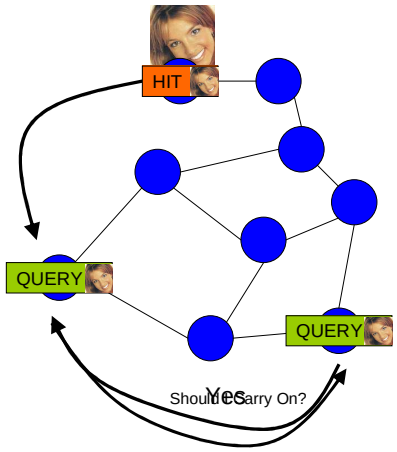
Query Routing (Keyword Hashing)

- Technique
  - Peers exchange keyword hash tables of the resources they share
  - QUERYs are forwarded to peers who most likely hold the resource
- Advantages
  - More direct searching eliminating broadcast traffic
- Disadvantages
  - Transient nature of users joining and leaving P2P network leads to out of date hash table references



Random Walks

- Technique
  - The QUERY (walker) is sent to only one randomly selected peer who in turn forwards it to one of its peers
  - Rather than use TTL, the walker reports back to its originator asking if it should continue through the network.
- Advantages
  - Traffic is directly proportional to the number of walkers per search (i.e. not exponential)
- Disadvantages
  - Longer delay receiving results



Clustering Techniques

- Technique
  - Nodes select peers that are topologically close to them organising into clusters.
- Advantages
  - If QUERYs can be satisfied locally then the underlying network is used efficiently to do that.
- Disadvantages

Summary

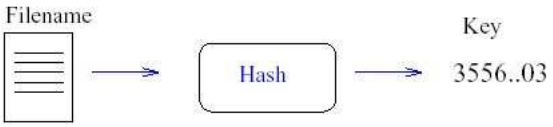
- We looked at
  - What P2P networks are
  - Gnutella
    - Original protocol
    - Current client optimisation techniques
  - Related unstructured P2P research
    - Searching for resources
    - Overlay network efficiency
- Concluding remarks
  - The original Gnutella protocol suffers from severe scalability issues due to message broadcasting
  - However, current research offers more scalable techniques for accomplishing both search and overlay construction in unstructured P2P networks which can be applied to new file sharing clients such as Gnutella

Protocolli P2P di seconda generazione

- Problema, i protocolli usati da Napster e Gnutella non sono scalabili;
- Per migliorare la scalabilità sono nati i cosiddetti protocolli P2P di seconda generazione che supportano DHT (Distributed Hash Table);
- Alcuni esempi di questi protocolli sono: Tapestry, Pastry, Chord, Can, Viceroy;

DHT

- A ogni file e ad ogni nodo è associata una chiave;
- La chiave viene di solito creata facendo l'hash del nome del file;
- Ogni nodo del sistema è responsabile di un insieme di file(o chiavi) e tutti realizzano una DHT;
- L'unica operazione che un sistema DHT deve fornire è lookup(key), la quale restituisce l'identità del responsabile di una determinata chiave.



DHT Routing

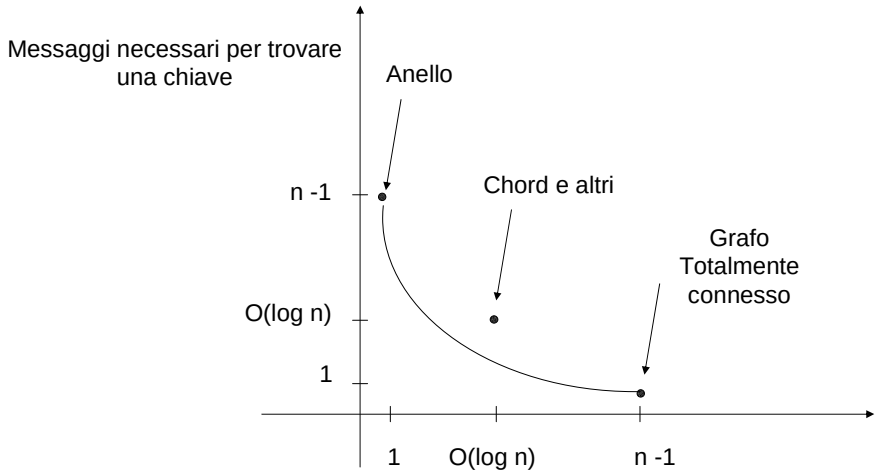
• La scalabilità di un protocollo è direttamente legata all'efficienza dell'algoritmo usato per il routing;

• In questo senso sostanzialmente gli obiettivi sono due:

  • Minimizzare il numero di messaggi necessari per fare lookup;

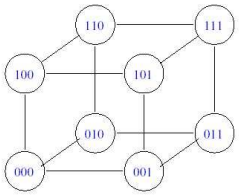
  • Minimizzare, per ogni nodo, le informazioni relative agli altri nodi;

• I vari DHT conosciuti differiscono proprio nel routing;



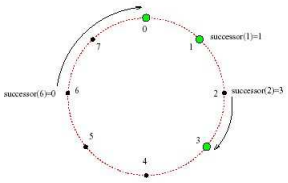
n è il numero dei peer;

DHT Routing: Tapestry



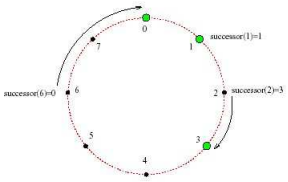
- Realizzazione dinamica dell'algoritmo di Plaxton et al.(che non si adattava a sistemi dinamici);
- Supponendo che le chiave è costituita da un intero positivo l'algoritmo di routing corregge a ogni passo un singolo digit alla volta;
- Per fare ciò un nodo deve avere informazioni sui nodi responsabili dei prefissi della sua chiave; (O(log N) nodi)
- Il numero di messaggi necessari per fare lookup è O(log N);
- L'algoritmo in pratica simula un Ipercubo;

DHT Routing: Chord



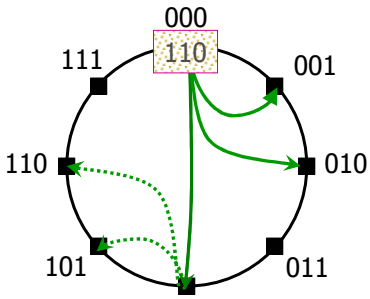
- Le chiavi sono mappati su un array circolare;
- Il nodo responsabile di una determinata chiave è il primo nodo che la succede in senso orario;
- Ogni nodo x di Chord mantiene due insiemi di vicini:
  - I log N successori del nodo x più il predecessore. Questo insieme viene usato per dimostrare la correttezza del Routing;
  - Un insieme log N nodi distanziati esponenzialmente dal nodo x, vale a dire l'insieme dei nodi che si trovano a distanza 2^i da x per i che va da 0 a log N – 1. Questo insieme viene usato per dimostrare l'efficienza del Routing;

DHT Routing: Chord

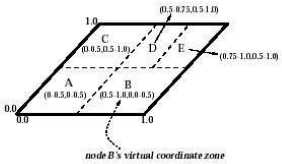


- Le informazioni che il nodo deve mantenere sugli altri nodi sono log N + log N + 1 = O(log N);
- Il numero di messaggi necessari per fare lookup è O(log N);
- Il costo che si paga quando un nodo lascia o si connette alla rete è di O(log2N) messaggi;
- L'algoritmo in pratica simula un Ipercubo, inoltre si comporta molto bene in un sistema dinamico;
  - Svantaggi:
  - una sola dimensione;
  - una sola strada;

DHT Routing: Chord



DHT Routing: CAN



- I nodi sono mappati su un toro d-dimensionale;
- A ogni nodo è associato un sottoinsieme di questo spazio d-dimensionale;
- Ogni nodo mantiene la lista dei nodi responsabili dei sottospazi che confinano con il proprio sottospazio;
- Ogni nodo ha O(d) vicini (due per ogni dimensione);
- Il routing avviene in $O(dN^{\frac{1}{d}})$ passi, $d/4 N^{\frac{1}{d}}$ in media ;
- Da notare che se usiamo d = log N dimensioni abbiamo O(log N) vicini e il routing ha costo:

$$O(\log N * N^{\frac{1}{\log N}}) = O(\log N * 2^{\log N^{\frac{1}{\log N}}}) = O(\log N * 2^{\frac{1}{\log N}\log N}) = O(\log N)$$

Riferimenti

- http://www.pdos.lcs.mit.edu/chord/
- http://www.napster.com/
- http://www.gnutella .com/
- http:// www.gnutella2.com/
- http:// www.shareaza.com/
- http://www.overnet.com/
- http:// www.openp2p.com/
- S. Ratnasamy, S. Shenker, and I. Stoica. *"Routing algorithms for DHTs: Some open questions"*. In In 1st International Peer To Peer Systems Workshop (IPTPS02).
- I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. Balakrishnan, *"Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications"*. In IEEE/ACM Trans. on Networking, 2003.

Domande?