

P2P: Overlay

Matteo Dell'Amico

Master SIIT
1 luglio 2008

Scaletta

- 1 Perché progettare sistemi P2P
 - Che significa “P2P”?
 - Vantaggi e svantaggi
 - Obiettivi
- 2 Breve storia del file-sharing
 - La prima generazione: Napster e Gnutella
 - Seconda generazione: Superpeer e Kazaa
 - I più usati (per ora): eMule e BitTorrent
- 3 Distributed Hash Table
 - Chord
 - Kademlia

Scaletta

- 1 Perché progettare sistemi P2P
 - Che significa "P2P"?
 - Vantaggi e svantaggi
 - Obiettivi
- 2 Breve storia del file-sharing
 - La prima generazione: Napster e Gnutella
 - Seconda generazione: Superpeer e Kazaa
 - I più usati (per ora): eMule e BitTorrent
- 3 Distributed Hash Table
 - Chord
 - Kademlia

Che intendiamo per "Peer-to-Peer"?

Da pari a pari?

- Implica **eguaglianza tra i nodi**: non del tutto preciso.

Pseudo-definizione

- Concetto principale: **decentralizzazione**.
 - In senso debole: **alcune parti** del sistema sono decentralizzate.
 - In senso forte: **tutto** il sistema è decentralizzato.

Che intendiamo per "Peer-to-Peer"?

Da pari a pari?

- Implica **eguaglianza tra i nodi**: non del tutto preciso.

Pseudo-definizione

- Concetto principale: **decentralizzazione**.
 - In senso debole: **alcune parti** del sistema sono decentralizzate.
 - In senso forte: **tutto** il sistema è decentralizzato.

Perché scrivere applicazioni P2P

- **Scalabilità:** quando aumenta la domanda di risorse, **anche l'offerta cresce.**
- **Resistenza a guasti ed attacchi:** non esistono "elementi centrali" la cui caduta **compromette il sistema.**
 - Si fa uso della **ridondanza.**
- **Risparmio:** vengono usate risorse "a costo zero".
 - Però, la ridondanza ha un costo...

Perché NON scrivere applicazioni P2P

- **Complessità:** progettare applicazioni P2P efficienti è molto difficile.
 - **Poche classi** di applicazioni P2P **vengono davvero usate.**
- **Churn:** non si può fare affidamento su una topologia stabile.
 - Bisogna tenere in conto che i "guasti" sono all'ordine del giorno.
- **Attacchi:** dobbiamo tenere presente che **chiunque potrebbe essere male intenzionato.**

Obiettivi

Desiderabili

- 1 Scalabilità
- 2 Disponibilità
- 3 Resistenza a guasti
- 4 Cooperazione tra i nodi

Opinabili

- 1 Anonimato
- 2 Resistenza a censura

Obiettivi

Desiderabili

- 1 Scalabilità
- 2 Disponibilità
- 3 Resistenza a guasti
- 4 Cooperazione tra i nodi

Opinabili

- 1 Anonimato
- 2 Resistenza a censura

Esempio: il file sharing

- 1 **Scalabilità:** più peer vogliono un file, più peer hanno il file (o sue parti).
 - 2 **Disponibilità:** se un peer si sconnette, possiamo continuare il download da un altro.
 - Se non si può, danno comunque non critico.
 - 3 **Resistenza a guasti:** dipende dall'overlay; **vedremo** (subito).
 - 4 **Cooperazione:** dipende dagli incentivi; **vedremo** (giovedì).
-
- 1 **Anonimato:** si può fare ma è costoso; **vedremo** (giovedì).
 - 2 **Resistenza a censura:** i fatti la dimostrano.

Scaletta

- 1 Perché progettare sistemi P2P
 - Che significa “P2P”?
 - Vantaggi e svantaggi
 - Obiettivi
- 2 Breve storia del file-sharing
 - La prima generazione: Napster e Gnutella
 - Seconda generazione: Superpeer e Kazaa
 - I più usati (per ora): eMule e BitTorrent
- 3 Distributed Hash Table
 - Chord
 - Kademlia

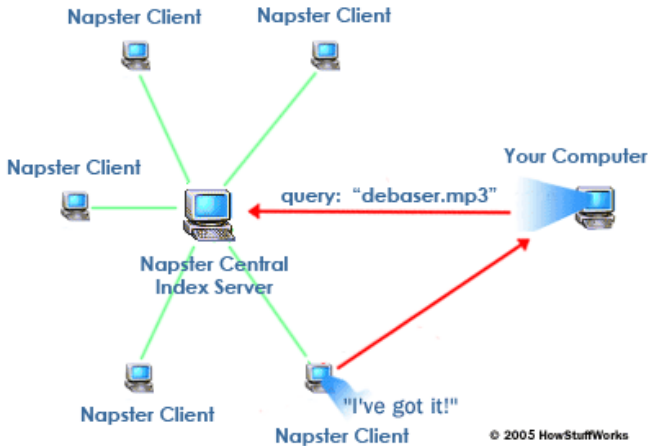
Napster



- Creato nel **1999** da Shawn Fanning, per trovare più facilmente musica online.
- Basato su un **server centrale**, con l'indice degli MP3 condivisi dagli utenti.
- Raggiunge 25 milioni di utenti registrati nel 2001.
- Nel luglio 2001, **viene chiuso** per un contenzioso legale perso.
- Evoluzioni del protocollo sopravvivono in **OpenNap** (WinMX).

Il protocollo di Napster

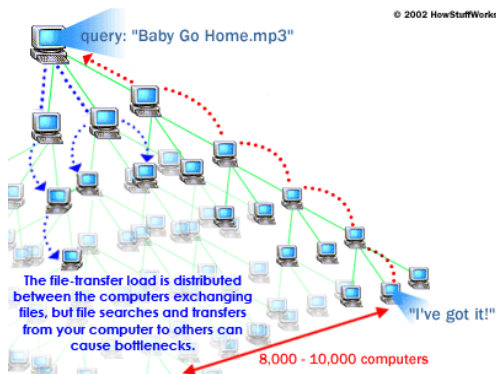
Napster Protocol



Gnutella

- Sistema **P2P puro**, senza componenti centralizzate sviluppato da Nullsoft (AOL).
- Pubblicizzato il 14 marzo 2000 su slashdot.org (“News for nerds”), **migliaia di utenti lo scaricano**.
- Il giorno successivo AOL ne **blocca il download**, ma la rete sopravvive.
- Vengono sviluppati **cloni** rilasciati come software libero.
- Ad oggi, il protocollo di Gnutella (riveduto e corretto) sopravvive in client come **LimeWire**.

Gnutella: protocollo originale



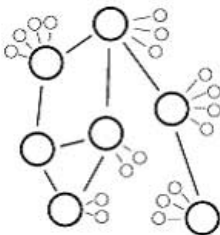
- Per evitare la propagazione indefinita delle richieste, si usa un parametro **TTL** che limita il numero degli hop (default 7).

Kazaa



- Creato da Niklas Zennström, Janus Friis e Priit Kasesalu (Skype, Joost) nel **marzo 2001**.
- I **superpeer** (o supernode, ultrapeer) nell'overlay (FastTrack) rendono l'applicazione la più efficiente.
 - Poi introdotti nella rete Gnutella.
- Il sistema più usato nel periodo 2001-2004.
- L'applicazione subisce cattiva pubblicità perché contiene **spyware** e **malware**.
- Tra il 2005 e il 2007, questioni giudiziarie portano alla **morte della rete**.

Superpeer



- Solo i nodi con maggiori risorse ed uptime vengono eletti come superpeer.
- Ogni nodo “regolare” comunica la propria lista di risorse al “suo” superpeer.
- Il compito della ricerca delle risorse viene limitato alla rete dei superpeer.

eMule



- Software libero basato sul protocollo di **eDonkey** (†2006).
- **Due** diversi overlay per la ricerca dei file:
 - **eDonkey**, con i server in stile Napster/OpenNap;
 - **Kad**, basato sulla Distributed Hash Table **Kademlia** (vedremo).
- **Hash SHA-1** per verificare l'autenticità dei file e di loro parti.
- **Source exchange**: i nodi che stanno scaricando lo stesso file si scambiano informazioni su chi sono gli altri.
- **Crediti** per incoraggiare l'upload (vedremo giovedì).

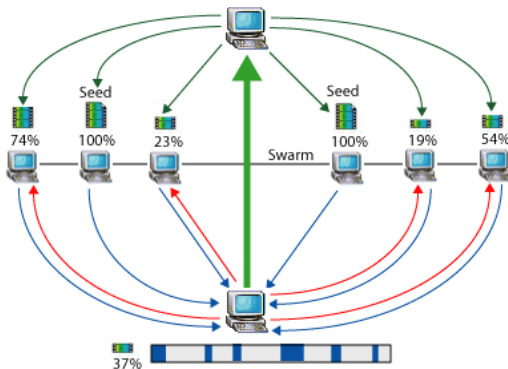
BitTorrent



- Il sistema che al momento **genera più traffico**.
- Concepito per lo **scambio legale di contenuto**, **non implementa la ricerca**.
 - Effettivamente usato sia per software libero che proprietario.
- È necessario ottenere (tramite mezzi esterni alla rete) un file .torrent che contiene l'indirizzo di un **tracker**.
- Il tracker fornisce gli indirizzi IP degli altri peer che scaricano il software.
- Sistema basato sulla **reciprocità** per incoraggiare il download (vedremo giovedì).

BitTorrent - il protocollo

BitTorrent tracker identifies the swarm and helps the client software trade pieces of the file you want with other computers.



Computer with BitTorrent client software receives and sends multiple pieces of the file simultaneously.

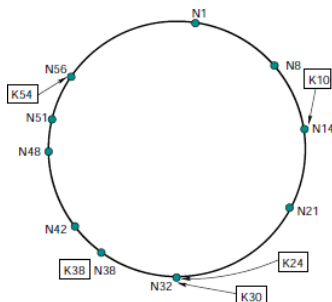
Scaletta

- 1 Perché progettare sistemi P2P
 - Che significa “P2P”?
 - Vantaggi e svantaggi
 - Obiettivi
- 2 Breve storia del file-sharing
 - La prima generazione: Napster e Gnutella
 - Seconda generazione: Superpeer e Kazaa
 - I più usati (per ora): eMule e BitTorrent
- 3 Distributed Hash Table
 - Chord
 - Kademlia

Distributed Hash Table

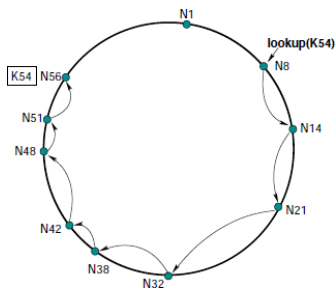
- Sistema scalabile basato sull'idea di creare una tabella hash **distribuita**.
- Ad ogni peer corrisponde uno slot della tabella hash.
- **Consistent hashing**: aggiungere o togliere slot ha un impatto piccolo sull'allocazione delle risorse nella tabella
 - Perfetto per gestire il churn.
- Esempio: nel file sharing, la DHT contiene - per ogni keyword k , l'indirizzo del peer che ha il file con questa keyword all'indirizzo $\text{hash}(k)$.

Chord - l'anello



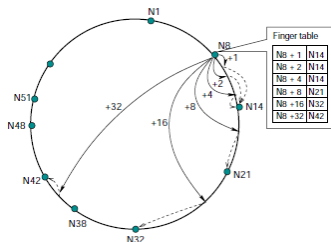
- Lo spazio di hashing (qui 0-63) viene mappato su un anello.
- Ogni nodo prende un identificatore a caso, e si inserisce nell'anello, con link al successore ed al predecessore.
- I documenti vengono inseriti sul primo peer con identificatore maggiore del codice hash relativo.

Chord - lookup



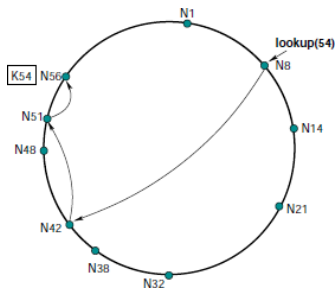
- Per inserire o cercare un oggetto, si possono percorrere i link “successore” finché non si arriva a destinazione.
- Metodo molto inefficiente.

Chord - i finger



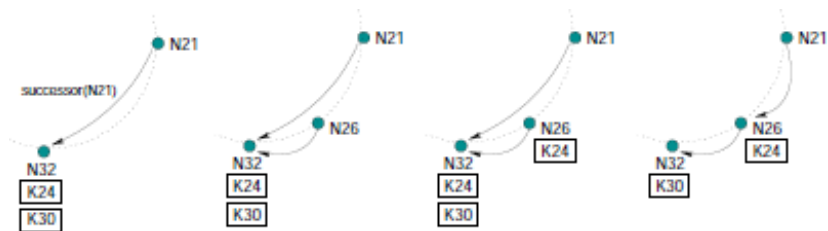
- Per arrivare prima a destinazione, si possono aggiungere salti a distanza di ogni potenza di 2.
- Il nodo NX può ottenere il finger "+Y" tramite:
 - Un lookup per "X+Y";
 - Chiedendo al finger "+Y/2" il suo finger "+Y/2".

Chord - greedy routing



- Se ad ogni passo usiamo il finger che ci avvicina di più alla destinazione **senza superarla**, saltiamo moltissimi passi.

Chord - nuovi nodi

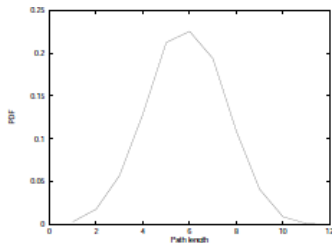


- L'inserimento di un nuovo nodo necessita di:
 - un lookup;
 - scambio di dati con soli due nodi.

Chord - fault tolerance

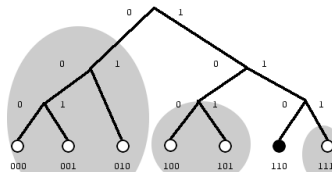
- Per evitare che la catena dei successori si spezzi quando i peer lasciano il sistema, si mantiene una lista degli r successori.
 - Se tutti gli r successori lasciano il sistema, c'è il rischio di spezzare la rete.
- Mantenimento periodico di tutti i link tramite la procedura di *stabilizzazione*.

Chord - numero di passi



- Numero medio di passi per arrivare a destinazione (qui, $n = 2^{12} = 4096$): $\frac{\log_2 n}{2}$.

Kademlia: cenni



- L'idea è la stessa: tenere liste di nodi via via più vicini: si ha un numero logaritmico di informazioni e un numero logaritmico di passi per arrivare a destinazione.
- In questo caso si mantengono delle “liste di finger” per nodi che hanno i primi $0, 1, \dots, k$ bit in comune (hash a k bit).

Per saperne di più...

- Wikipedia (più informazioni nelle pagine in inglese): Napster, Gnutella, Kazaa, FastTrack, eMule, BitTorrent, Kademlia.
- Articolo di Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications
<http://pdos.csail.mit.edu/papers/ton:chord/>.
- Fatemi domande :-)