ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SECONDA FACOLTA' DI INGEGNERIA - SEDE DI CESENA
Corso di Laurea in Ingegneria Informatica

Streaming audio e video nei sistemi Peer-To-Peer TV: il caso SopCast P2PTV

Tesi di Laurea in Sistemi Distribuiti

Relatore:
Prof. Ing. Andrea Omicini

Presentata da:
Riccardo Timoncini

Sessione I
Anno Accademico 2011/2012

Alla mia famiglia, cui devo tutto.

A Gessica che mi è stata vicina nei momenti difficili.

Agli amici, parenti, conoscenti, cani, alberi, libri, stanze,

strade, auto e a qualsiasi persona, animale o cosa

che ha sopportato i miei momenti pesanti

in questi anni di università.

A tutti voi, Grazie!

Indice

Elenco delle figure

Introduzione

Negli ultimi anni lo streaming peer-to-peer (P2P) di contenuti multimediali è diventato un tema molto dibattuto dai ricercatori e sempre più conosciuto anche da centinaia di milioni di utenti finali che godono gratuitamente della possibilità di trasmettere canali tv, radio o condividere film e file. Il motivo è che è facile da usare e molto più veloce e robusto rispetto al tradizionale sistema di server-client. Supporta eterogeneità molto diverse in materia di ampiezza di banda, di storage e di rete, rendendolo così accessibile a un vasto pubblico.

In una rete peer-to-peer gli end-point si organizzano in una sovrapposizione a livello di applicazione e scambiano i dati tra di loro. Questi sistemi sono robusti, facili da implementare e riducono il carico della sorgente di contenuti, in quanto il costo di distribuzione, in termini di larghezza di banda e potenza di elaborazione è condivisa dai nodi grazie alla struttura a sovrapposizione (overlay network).

Ci sono però ancora molte questioni aperte su quei sistemi su larga scala. Al fine di supportare i requisiti di ritardo, i fallimenti di accoglienza, l'accesso e l'abbandono degli utenti che introduce forte dinamicità, l'architettura P2PTV si dovrebbe occupare di questi problemi. Tuttavia la maggior parte dei sistemi video disponibili in streaming soffrono ancora a causa di ritardi in trasmissione, interruzioni, discontinuità di

riproduzione, mancanza di qualità del servizio e molto spesso anche algoritmi di selezione dei peer troppo approssimativi che non supportano la scalabilità del sistema. Il mio progetto è suddiviso in 4 capitoli.

Nel primo verrà introdotto il tema principale della tesi: dopo un breve ricorso storico riguardo alla nascita dello streaming e al suo sviluppo, verranno descritte le caratteristiche e il protocollo di comunicazione client-server utilizzato, verranno proposti i modelli più diffusi per i sistemi basati sullo streaming p2p mettendo in evidenza l'importanza della qualità del servizio

Il secondo capitolo affronta il tema della Qualità del Servizio dei sistemi p2p utilizzati per il live streaming. Si introducono i parametri metrici riguardanti il mondo p2p in genere per poi soffermarci sugli aspetti che riguardano le applicazioni come le P2PTV che dunque forniscono contenuto multimediale in streaming; senza tralasciare quelli che possono essere gli aspetti sui quali intervenire per migliorare la QoS.

Il terzo capitolo affronta il tema della performance dei sistemi p2p partendo dai criteri metrici attraversi i quali viene valutata. Risulta di primaria importanza trattare il principio di Equità e prestare attenzione anche all'identificazione dei nodi stabili che rendono il servizio più affidabile.

Nel quarto ed ultimo capitolo viene presentato uno dei principali esempi di applicazione P2PTV chiamata SopCast. Prima di addentrarci nell'analisi dell'architettura software verrà fatta una breve introduzione circa le Peer-To-Peer Television, evidenziandone peculiarità, caratteristiche strutturali e modelli di riferimento. Verrà trattato il protocollo BitTorrent sul quale, secondo molti ricercatori, SopCast sia fortemente improntato e successivamente elencate le principali client applications per

trasmissione di canali TV su reti p2p. Tratterò poi SopCast, sviluppo e caratterizzazione dell'interaccia dal punto di vista chi apre un canale di trasmissione e di chi invece fruisce dei suoi contenuti. Seguirà un elenco di esperimenti condotti da un team di ricercatori cinesi che hanno analizzato il protocollo nello specifico, identificando le tipologie di pacchetti che viaggiano tra i nodi connessi e la struttura a tre livelli di sovrapposizione. Concluderemo con delle considerazioni sulle possibili implicazioni che la natura dinamica del sistema SopCast apporta a livello topologico.

Capitolo 1

Lo streaming tra passato, presente e futuro

In questo capitolo verrà introdotto il tema principale della tesi. Nel primo paragrafo verrà fatto un ricorso storico riguardo alla nascita dello streaming e al suo sviluppo, nel secondo paragrafo verranno descritte le caratteristiche e il protocollo di comunicazione client-server utilizzato, nel terzo paragrafo verranno proposti i modelli più diffusi per i sistemi basati sullo streaming p2p, in particolare la rete strutturata e non strutturata.

1.1. Cenni storici

A partire dalla metà degli anni '80 i computer si sono enormemente sviluppati e potenziati ed hanno reso possibile la visualizzazione di una gran quantità di formati audio e video con qualità sempre maggiore. Mentre nei decenni Novanta era consuetudine al fine di visualizzare un contenuto multimediale remoto scaricare tutto un file da un server remoto, negli anni successivi, grazie principalmente al grande successo di Internet, alla creazione di protocolli per il web più specializzati e

all'utilizzo di una larghezza di banda maggiore, si sono verificati enormi cambiamenti in questo campo. Oggigiorno stanno nascendo nuove tecnologie che stanno prendendo piede anche grazie alla loro semplicità di implementazione sulle macchine e disponibilità ai singoli clienti della rete. Stiamo parlando ad esempio del Live Streaming P2P, delle radio e della TV su Internet, il cui obiettivo è la massa.

Lo streaming è sostanzialmente una ricezione di un flusso di dati audio e video risiedenti su un server da parte di un client. La peculiarità sta nel fatto che questi dati possono essere riprodotti in maniera continuativa man mano che giungono a destinazione.

Il primo evento diffuso tramite la tecnica dello streaming (1) è databile 18 novembre 1994, giorno in cui la famosa rock band statunitense The Rolling Stones si esibì in un concerto al Cotton Bowl di Dallas davanti a 50'000 fans. La Band in quell'occasione decise di rendere disponibile a scopo promozionale i primi 20 minuti dello spettacolo che diventò così il primo concerto in multicast del cyberspazio. Si ricordano le parole del cantante Mick Jagger che iniziò il concerto così: "I wanna say a special welcome to everyone that's, uh, climbed into the Internet tonight and, uh, has got into the M-bone. And I hope it doesn't all collapse!" [Vorrei dare un benvenuto speciale a tutti coloro che ci stanno seguendo da Internet questa sera e coloro che vengono dal M-bone. E spero che non collassi tutto!]. L'infrastruttura software che permise di fatti la visualizzazione dell'evento in diretta fu M-bone, una piattaforma virtuale fondata nel 1992 leader nella gestione di audio/video conferenze e spazi di comunicazione condivisa.

A cavallo degli anni 2000 la maggioranza dei contenuti fruibili via streaming si trovavano in formato Real Video[1], che necessitava della presenza sull'elaboratore di un software applicativo per visualizzare il contenuto multimediale. In Italia i programmi che diffondo in tempo reale contenuti video sono diventati discretamente popolari nel 2006 durante i Campionati Mondiali di Calcio. Avendo la RAI acquistato solo alcune partite del campionato molte persone utilizzavano questa tipologia di programmi che appoggiandosi a server in paesi esteri (spesso cinesi) permettevano di visionare la partite con una qualità video accettabile.

Col passare degli anni a partire dall'avvento di CoolStreaming[2], il progenitore di tutti i P2PTV[3], si sono moltiplicati i programmi che offrono questo servizio. Ai giorni d'oggi infatti i principali software sono davvero tantissimi: SopCast, PPlive, PPStream, TVAnt, Zattoo, Joost, Tvkoo, GridMedia, Vudu, Tribler, Octoshape, Roxbeam. Utilizzando questa tecnologia le grandi aziende nell'ambito dei media e della comunicazioni stanno sperimentando sempre migliori tecniche per fornire contenuti a pagamento tramite tecnologie peer-to-peer avendo come obiettivo di massimizzare la qualità di invio (dei dati) ad ogni singolo utente malgrado l'eterogeneità e l'asimmetria delle capacità in banda di un sempre maggior numero di partecipanti alla rete.

---

[1] Real Video è un codec video proprietario, sviluppato dalla RealNetworks a partire dal 1997. È stato usato inizialmente per servire video streaming attraverso le reti internet ad un basso bit rate verso personal computer. Real Video differisce dalle codifiche normali in quanto è un formato proprietario ottimizzato esclusivamente per lo streaming attraverso il protocollo (proprietario) PNA oppure tramite il Real Time Streaming Protocol.

[2] CoolStreaming è un aggregatore di televisioni online, ovvero un programma che raggruppa sotto diverse categorie i riferimenti diretti a flussi audiovisivi provenienti dai vari broadcaster. Coolstreaming nasce ufficialmente nel marzo del 2005 come supporto al primo programma P2PTV (Coolstreaming Cinese), poi tramutato in un portale IPTV/NET-TV.

[3] P2PTV è una tecnologia peer-to-peer sviluppata per trasmettere flussi video, tipicamente canali televisivi.

1.2. Le caratteristiche dello streaming

Per specificare gli aspetti caratteristici dello streaming si rende necessario in primis effettuare una distinzione tra due tipologie di streaming: il live streaming e lo streaming on demand.

Nella primo caso ogni cliente comunica con gli altri per riuscire ad ottenere il file che ricerca. Il file non può essere salvato su hard disk poiché il suo contenuto è disponibile una sola ed unica volta, ovvero nel momento in cui il video/audio viene trasmesso. La comunicazione tra peer prevede la condivisione di parti del file in maniera da aumentare il througput dell'applicazione.

Nel secondo caso, ovvero per quanto concerne lo streaming on demand, i client effettuano richieste direttamente al server nel quale risiede fisicamente il file da condividere. A differenza del live streaming, qui è possibile salvare il file su disco e visionarlo poi successivamente quando lo si desidera. Infatti quando il cliente richiede un contenuto ad una sorgente, sarà questa ad occuparsi di creare un canale comunicativo con l'utente e ad effettuare l'invio del file, il quale per incominciare ad essere visualizzato non necessita del download completo da parte di colui che lo richiede. Si potranno infatti incominciare a visualizzare le parti ricevute già dopo pochi istanti dall'inizio del caricamento. Entra quindi in gioco il fattore tempo che è quell'elemento che differenzia le due modalità: mentre nel live streaming il tempo assume molta importanza poiché la cooperazione tra peer può avvenire soltanto in seguito alla trasmissione dal server o in momenti successivi, ma mai a trasmissione conclusa, nello streaming on demand questo aspetto è assolutamente trascurabile. Ciò che invece accomuna le due tipologie è il fatto che entrambe introducono un ritardo nella visualizzazione delle parti di video che sono state ricevute per proteggere la riproduzione da eventuali problemi causati dalla connessione di rete.

1.2.1. Il protocollo RTSP

Il protocollo utilizzato per lo streaming nel modello ISO/OSI[4] a livello applicazione è l'RTSP, ovvero Real Time Streaming Protocol (2). In realtà bisogna citare anche altri protocolli poiché l'RTSP non si occupa in primis della trasmissione dati, che invece spetta al protocollo RTP[5], acronimo di Real Time Transport Protocol, e ai protocolli TCP[6], Trasmission Control Protocol o UDP[7], User Data Protocol.

Il RTSP è stato sviluppato da RealNetworks, Netscape Communicaitons e Comumbia University di New York e vide la sua pubblicazione nel 1998 come RFC 2326. È stato progettato per essere simile al protocollo di rete HTTP[8], ma con alcune differenze (3):

- Un server RTSP ha la necessità di mantenere delle informazioni di stato, in contrapposizione al funzionamento di HTTP che è privo di un concetto di stato.

  HTTP è definito *stateless* perché il server non ha l'esigenza di memorizzare lo stato della connessione

[4] ISO/OSI è uno standard per reti di calcolatori stabilito nel 1978 dall'International Organization for Standardization, il principale ente di standardizzazione internazionale, che stabilisce per l'architettura logica di rete un'architettura a strati composta da una pila di protocolli suddivisa in 7 livelli: fisico, datalink, rete, trasporto, sessione, presentazione, applicazione.

[5] RTP è un protocollo del livello applicazione utilizzato per servizi di comunicazione in tempo reale su Internet e viene più spesso impiegato in applicazioni unicast.

[6] TCP è un protocollo orientato alla connessione, ovvero prima di poter trasmettere dati deve stabilire la comunicazione, negoziando una connessione tra mittente e destinatario, che viene esplicitamente chiusa quando non più necessaria. Questo protocollo si occupa anche del controllo sul flusso di byte che ha ricevuto il destinatario, il quale deve ricevere tutti i segmenti di dati una volta sola.

[7] UDP è un protocollo orientato alla trasmissione di pacchetti che non necessita di instaurare una connessione e quindi supporta la trasmissioni con più client. Non si occupa della ritrasmissione dei pacchetti persi o del riordinamento di questi una volta giunti a destinazione.

[8] HTTP acronimo di HyperText Transfer Protocol è un protocollo di trasferimento di un ipertesto e viene usato come principale sistema per la trasmissione d'informazioni sul web.

con un determinato client. Ad ogni modo per avere informazioni di sessione viene utilizzata la tecnica dei cookie.

Nel caso di RTSP utilizzando i protocolli TCP e UDP il server deve essere in grado di mantenere lo stato si sessione per permettere la corretta esecuzione della richiesta di un client.

- I dati sono portati fuori banda da un protocollo differente.

  HTTP è un protocollo asimmetrico e *in-band*. Il client esegue una richiesta al server ovvero attraverso al browser esegue una richiesta di una pagina web e il server risponde restituendo la pagina se questa si trova nel server, altrimenti una pagina di errore (Error 404 – Page Not Found). Sono contemplate richieste sono client → server.

  RTSP invece è un protocollo *out-of-band* poiché una parte del contenuto informativo come i dati dello streaming sono contenuti nell'in-band stream, mentre il contenuto informativo che riguarda i messaggi di controllo, che non fanno parte del contenuto principale, vengono definiti out-of-band.

  IN RTSP sia client che server possono inviare richieste, quindi sono possibili richieste client → server e server → client.

- RTSP usa per definizione lo standard ISO 10646 (UTF-8)[9] invece che l' ISO 8859-1, in modo da essere

[9] ISO 10646 (UTF-8) è di fatto lo standard universale per la rappresentazione dei caratteri.

consistente con le nuove caratteristiche di internazionalizzazione che si stanno introducendo nell'HTTP.

- RTSP utilizza sempre un URI (Universal Resource Identifier) delle richieste assoluto.

  HTTP pone nelle richieste solo un percorso assoluto, e inserisce il nome dell'host in un campo intestazione separato.

  RTSP utilizza sempre l'URI assoluto che permette di implementare in modo semplice il "virtual hosting", con cui si possono mantenere su un singolo host con un unico indirizo IP diverse strutture di documenti.

- RTSP introduce un certo numero di metodi nuovi e utilizza differenti identificatori di protocollo.

  I metodi HTTP sono: HEAD, GET, PUT, POST, DELETE, OPTION, CONNECT, TRACE.

  I metodi RTSP sono: DESCRIBE, ANNOUNCE, SETUP, GET_PARAMETER, SET PARAMETER, REDIRECT, OPTIONS, PAUSE, PLAY, RECORD, TEARDOWN.

Descrizione dei metodi (4):

DESCRIBE: il client usa questo metodo per ottenere una descrizione completa della risorsa, identificata tramite l'URL nella richiesta. In genere il client richiede un DESCRIBE per poter inizializzare il playback di una sessione.

ANNOUNCE: il client usa questo metodo quando vuole registrare sul server una nuova sessione, per informarlo di tutti i dettagli riguardanti la sessione, necessari per eseguirne con successo la registrazione (Client → Server).
Il server usa questo metodo quando vuole aggiornare in tempo reale la descrizione della sessione (Server → Client).

SETUP: il client, dopo aver ottenuto la descrizione di una sessione con un DESCRIBE, oppure averla fornita al server con un ANNOUNCE, deve inviare una richiesta di SETUP per ogni media stream compreso nella sessione di cui vuole fare il playback o il recording.

GET_PARAMETER: questo metodo può essere usato per chiedere informazioni sul valore di parametri del server o del client.

SET PARAMETER: questo metodo può essere usato per assegnare dei valori ad opportuni parametri riferiti ad una sessione o ad un singolo media stream.

REDIRECT: questo metodo informa il client che deve connettersi ad un altro server.

OPTIONS: il server implementa questo metodo per rispondere ad un client che voglia sapere quali

metodi supporta. Il server nella risposta inserisce un header con l'elenco dei metodi da esso supportati.

PAUSE: il client usa questo messaggio per chiedere al server una pausa nella riproduzione o nella registrazione della sessione.

PLAY: il client invia una richiesta di questo tipo quando vuole che il server cominci ad inviare i flussi multimediali, secondo le specifiche di inizializzazione impostate nel SETUP. Se la sessione è di tipo aggregato[10], il client deve inviare un solo PLAY per richiedere la trasmissione di tutti i media stream che la compongono. Se la sessione è di tipo non aggregato, il client deve invece inviare un messaggio di PLAY per ogni media stream che la compone.

RECORD: il client usa questo messaggio per chiedere al server di iniziare a registrare i media stream della sessione appena inizializzata tramite la fase di SETUP

TEARDOWN: questo messaggio è utilizzato dal client per chiedere la chiusura di una sessione: se la sessione è aggregata basta un solo messaggio di questo tipo; altrimenti il client deve chiedere il

[10] Controllo Aggregato: il controllo contemporaneo di flussi multipli. Ad esempio per un flusso audio ed uno video logicamente correlati, un client RTSP può inviare al server una singola richiesta di "play" o "pause", che avrà effetto su entrambi.

TEARDOWN per ogni media stream. Inoltre il server chiude la sessione e si disconnette dal client.

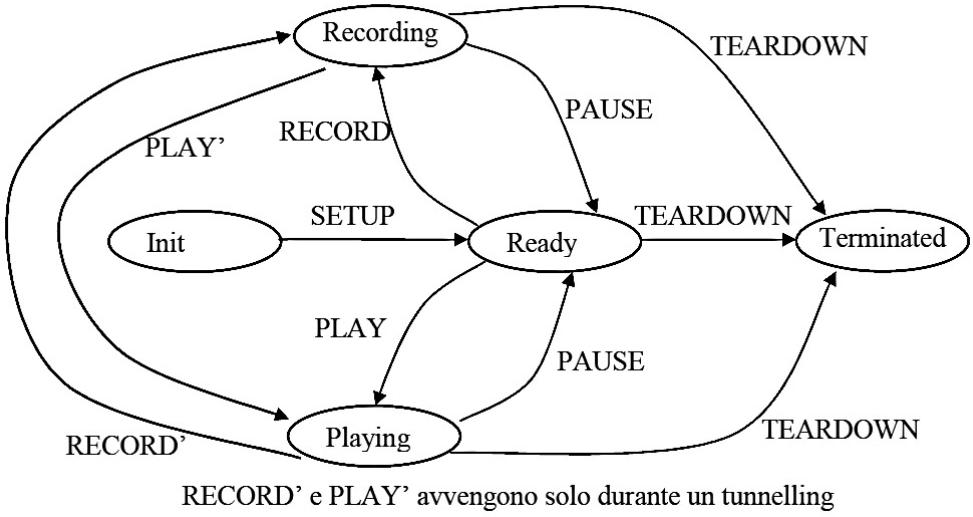


Figura 1.1: Automa del server e del client nel protocollo RTSP

La comunicazione tra client e server avviene in una prima fase tramite l'attivazione del protocollo HTTP, ossia il client effettua una richiesta di attivazione di una nuova sessione perché è interessato a scaricare un certo file in streaming.
Quando il web server risponderà al client sarà possibile eseguire il setup della sessione tramite RTSP e comunicare con un media server, nel quale sono memorizzati i dati multimediali. Appena il client avrà ricevuto una piccola parte del file, potrà incominciare a visualizzarla nel proprio elaboratore tramite un qualsiasi player video, potrà mettere in pausa la comunicazione senza perdere la possibilità di poter ripristinare il processo di download dal server.

Infine quando il processo di download e visualizzazione del file saranno terminati, si andrà a chiudere la comunicazione tra client e server liberando così tutte le risorse occupate fino a quel momento.

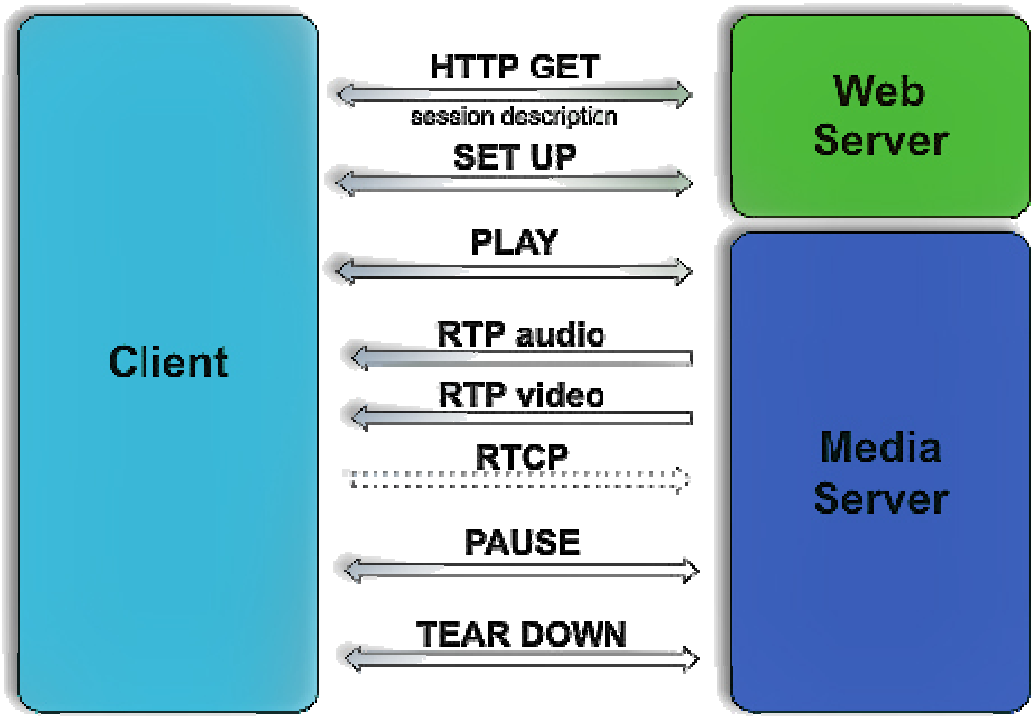


Figura 1.2: Comunicazione tra client e web/media server nel protocollo RTSP

1.2.2. Le modalità di diffusione dei dati

Si possono evidenziare quattro principali modalità per la diffusione dei dati in streaming:

Unicasting: tipologia di comunicazione che prevede la trasmissione di un messaggio da una sorgente ad un singolo destinatario. Tale connessione client-server è detta anche uno-ad-uno.

Broadcasting: tipologia di comunicazione che prevede la trasmissione di un messaggio da una sorgente a tutti gli ascoltatori connessi e pronti a riceverlo. Questa modalità ha come effetto collaterale che se il segmento da inviare è di medie o grandi dimensioni il rischio è quello di congestionare rapidamente la rete.

Multicasting: tipologia di comunicazione che prevede la trasmissione di un messaggio da una sorgente ad un gruppo di destinatari che possiedono un indirizzo univoco, detto indirizzo di multicast. Non vi è relazione diretta tra client e server. Il server prepara una sola risposta e la invia all'indirizzo di multicast. Saranno poi i router a diramare e replicare a tutti i clienti ascoltatori il contenuto del file da ricevere. Può essere il server a scegliere l'indirizzo e la porta di destinazione (come Live Streaming e Streaming-on-demand), ma potrebbe essere anche il client a specificare l'indirizzo e la porta. Tale connessione è detta anche uno-a-molti.

Peer-to-peer: è una rete di computer o in generale una qualsiasi rete che non possiede client o server fissi, ma un numero di nodi equivalenti (*peer*, appunto) che fungono sia da client che da server verso altri nodi della rete. Questo modello di rete è l'antitesi dell'architettura client-server. Mediante questa configurazione qualsiasi nodo è in grado di avviare o completare una transazione. I nodi equivalenti possono differire nella configurazione locale, nella velocità di elaborazione, nella ampiezza di banda e nella quantità di dati memorizzati. L'esempio classico di p2p è la rete per la condivisione di file noto come *File Sharing*. Il file sharing, che tradotto in italiano

indica la condivisione dei file, è l'idea di base della maggior parte delle reti p2p (Gnutella, eMule, BitTorrent, Kazaa). Molte di queste reti possono permettere di individuare più copie dello stesso file nella rete, di riprendere lo scaricamento interrotto di un file, di eseguire lo scaricamento da più fonti contemporaneamente riducendo i tempi di ritardo dovuti al download da singolo punto. E' proprio questa grande capacità di "facile reperimento di file di grande interesse" che ha permesso e ancora spinge lo sviluppo di questa tecnologia, portando allo sviluppo di nuove applicazioni. Prodotti più innovativi prevedono l'utilizzo delle reti peer-to-peer per la diffusione di elevati flussi di dati generati in tempo reale come è il caso delle P2PTV (Peer-To-Peer Television) e delle WebRadio. Questi programmi si basano sull'utilizzo contemporaneo della banda di ricezione (o downlink) e di trasmissione (o uplink) di cui dispone ciascun utente, in modo tale da trasmettere agli altri partecipanti alla rete p2p il flusso dati.

Questa tipologia di programmi, in linea di principio, non richiede l'utilizzo di:

- *server dotati di elevate prestazioni* dato che il server che genera il traffico multimediale fornisce a un numero molto limitato di utenti i flussi video e poi i singoli utenti ridistribuiscono agli altri utenti i flussi video. Questo metodo di diffusione permette in teoria la trasmissione in tempo reale di contenuti video ma richiede che i singoli utenti siano dotati di connessioni che oltre ad avere elevata banda in ricezione abbiano un'altrettanto elevata banda in trasmissione.

- *server di grandi dimensioni* per gestire molti utenti dato che se la rete è ben bilanciata si autosostiene e quindi è fortemente scalabile, con la possibilità di ottenere buoni guadagni con costi limitati. Mediante semplici programmi per l'analisi dei protocolli come Wireshark[11] (5), si possono studiare i vari parametri di interesse in una sessione Live Streaming per applicazioni di P2PTV o WebRadio al fine di apprendere il funzionamento di queste applicazioni direttamente dalla pratica.

Esempi di sistemi Peer-to-peer sono dunque i sistemi di condivisione e diffusione di informazioni come Wikipedia, i sistemi di condivisione file come BitTorrent, i sistemi di condivisione del calcolo come il progetto SETI@home[12], i sistemi per la privacy o la sicurezza come FreeNet[13], i sistemi di telefonia come la rete di Skype (6).

[11] Wireshark: è un software per analisi di protocollo o packet sniffer (letteralmente *annusa-pacchetti*) utilizzato per la soluzione di problemi di rete, per l'analisi e lo sviluppo di protocolli o di software di comunicazione e per la didattica.

[12] SETI@home: è un esperimento scientifico che utilizza i computer connessi ad Internet per la Ricerca di Intelligenze Extraterrestri, consultabile al sito web http://setiathome.berkeley.edu/

[13] Freenet: è una piattaforma su cui poter condividere file, chattare sui forum e pubblicare in forma anonima e senza timore di blocchi o censure, consultabile al sito web https://freenetproject.org/

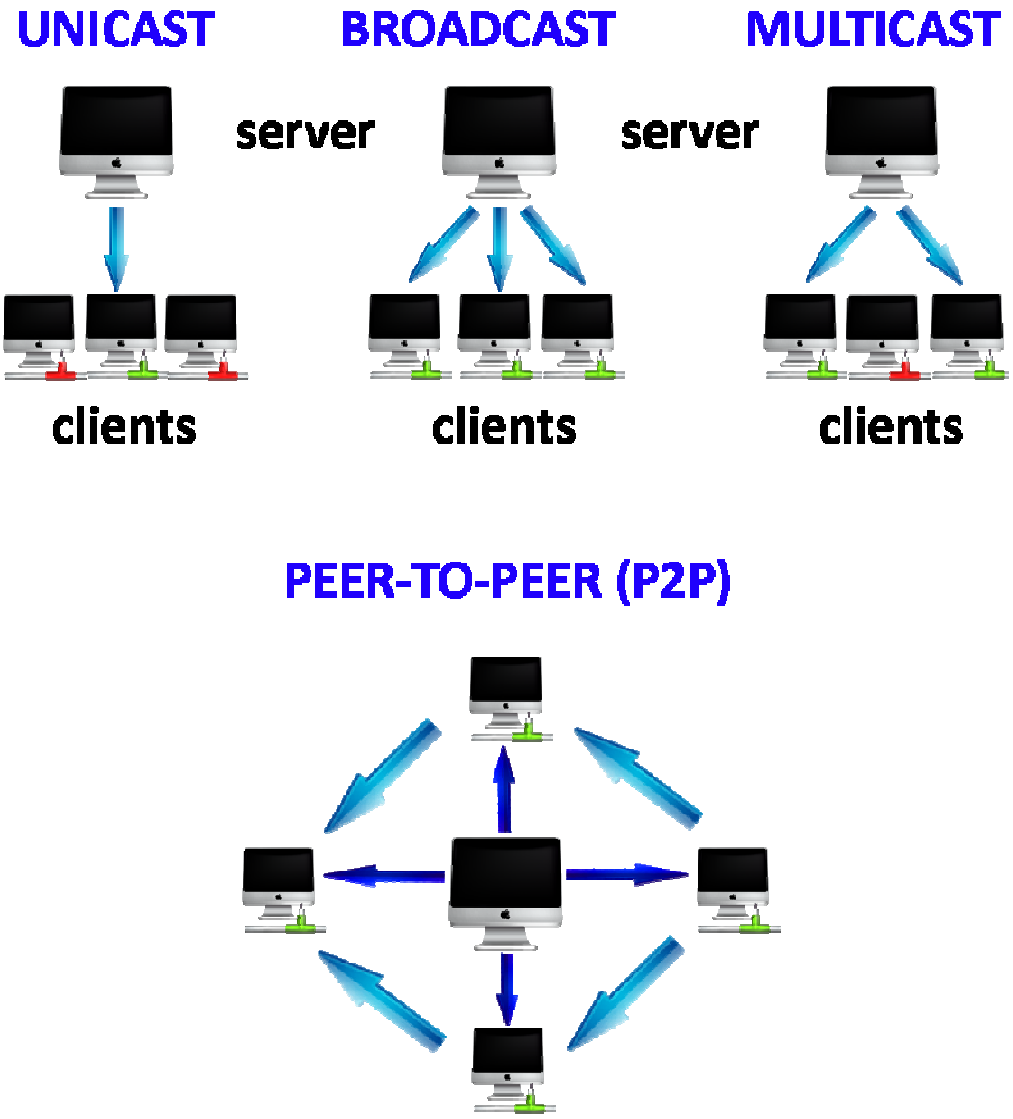


Figura 1.3: Principali modalità di diffusione dei dati.

1.3. Modelli per lo streaming peer-to-peer

Dopo avere introdotto nel paragrafo precedente quali sono le principali modalità di diffusione dei dati, si può ora notare come il modello preferibile sia proprio quello che

prevede la comunicazione di tutti i peer tra di loro evitando il collo di bottiglia introdotto dai sistemi multicast.
Nei multicast system infatti la presenza di un unico server possessore dei file causa delle difficoltà qualora aumenti il numero degli utenti che richiedono tale file, andando man mano a congestionare la rete. Soluzione a questo problema è stata appunto fornita dall'utilizzo di server multipli che hanno aumentato la scalabilità del sistema, la tolleranza ai guasti e la larghezza di banda.

Altro aspetto positivo dei sistemi p2p è che ogni nodo può entrare nella rete e abbandonarla quando vuole. Tuttavia potremmo incappare in una criticità qualora un peer stia scaricando vari *chunks*, parti di file, da un altro peer e quest'ultimo improvvisamente decida di lasciare la rete, interrompendo inaspettatamente il download.

Un ultimo aspetto da considerare è che nei sistemi p2p un client e un server possono essere suddivisi fisicamente in parti logicamente equivalenti che operano sull'insieme completo dei dati, ottenendo così un buon bilanciamento del carico. A differenza delle applicazioni client-server, come ad esempio unicast e multicast, in cui l'architettura ha una distribuzione dei componenti verticale, nei sistemi basati su peer la distribuzione è di tipo orizzontale (7).
Dato questo comportamento simmetrico, le architetture peer-to-peer si sviluppano attorno al problema dell'organizzazione dei processi in una *rete overlay* (overlay network), ovvero una rete in cui i nodi sono costituiti dai peers e i collegamenti rappresentano i possibili canali di comunicazione (che di solito sono realizzati mediante connessioni TCP). In generale un processo non può comunicare direttamente con un altro

processo arbitrario, ma deve inviare messaggi attraverso i canali di comunicazione disponibili.

Esistono due tipi di reti overlay: quelle strutturate e quelle non strutturate. Per quanto riguarda le prime, i peer utilizzano una specifica strategia per comunicare tra di loro e utilizzano una struttura ad albero, a multi-alberi o una rete mesh (come ad esempio una DHTs[14]).

Per quanto riguarda le seconde invece i peer utilizzano un particolare protocollo, detto Protocollo Gossip, che rende più libera la comunicazione tra peer, nel senso che un nodo può scegliere a quale nodo allacciarsi nel momento che desidera.

1.3.1. Rete strutturata

Architettura ad albero

La struttura è organizzata a livelli: nel livello più in alto risiede il padre, ovvero la radice dell'albero che è la sorgente del file da trasmettere via via ai livelli sottostanti dove si trovano i nodi figli. Le parti che compongono il file, i chunk, a partire dal padre si dirameranno fino a giungere ai nodi foglia dell'albero permettendo così la completa visualizzazione del file.

Un aspetto di criticità che si potrebbe presentare utilizzando questo tipo di architettura è la situazione in cui alcuni nodi interni all'albero decidessero di abbandonare la rete (8). Tale problema è denominato node churn e come detto è causato dal comportamento dinamico dei nodi della rete e può

[14] DHTS (Distributed Hash Tables): sono tabelle di hash distribuite che costituiscono una classe di sistemi distribuiti decentralizzati che partizionano l'appartenenza di un set di chiavi tra i nodi partecipanti, e possono inoltrare in maniera efficiente i messaggi all'unico proprietario di una determinata chiave. Le DHT sono tipicamente progettate per gestire un vasto numero di nodi e possono essere utilizzate per implementare servizi più complessi, quali file system distribuiti, sistemi peer-to-peer di file sharing, web, caching cooperativo e domain name services.

portare ad un degrado della qualità del contenuto da diffondere. Attualmente una ottima proposta di soluzione è quella di tentare in scale temporali brevi di effettuare un aggiornamento dei peer coinvolti e un ribilanciamento in termini di efficienza, stabilità di connessione e disponibilità dei contenuti. È stato dimostrato che questo approccio porta ad un miglioramento uniforme della topologia distribuita e a migliori prestazioni grazie al ridimensionamento del node churn (9).

Architettura a multi-alberi

Una evoluzione della precedente topologia sono i multi-alberi, in cui i nodi interni di un albero possono essere nodi foglia di un altro albero garantendo così un'instradamento dei dati più efficace e una migliore distribuzione delle informazioni. In questa architettura in base alle caratteristiche della propria connessione di rete ogni peer decide a quanti alberi appartenere in maniera tale da massimizzare le connessioni con altri peer (10).

Architettura a rete mesh

I vari problemi legati alla struttura ad albero hanno portato allo sviluppo di una nuova strategia in cui i peer partecipanti formano una struttura a maglia (*mesh based structure*) e incorporano una capacita di invio SWARM-LIKE, ispirata dal meccanismo noto come file-swarming (sciame di file) utilizzato ad esempio nella più diffusa applicazione per la condivisione di file nelle reti p2p, ovvero BitTorrent (11).

Sono due i concetti base della struttura a maglia e sono la *Overlay Construction* (costruzione dell'infrastruttura di rete) e la *Content Delivery* (consegna del contenuto).

- Costruzione dell'infrastruttura di rete: in un meccanismo di Streaming P2P mesh-based i peer partecipanti formano una struttura a maglia in cui le connessioni tra i partecipanti e la direzione dell'invio di dati non è regolata da un rapporto fisso di parentela *padre-figlio* come nel caso delle reti con topologia ad albero o a foresta. Fatta eccezione per la sorgente, in questa struttura ogni peer è sia padre che figlio.
- Consegna dei contenuti: la consegna dei dati tra i vari peer è ottenuta mediante "la spinta" (push reporting) del flusso media da parte di alcuni peer insieme con la richiesta di ottenimento dati (pull requesting) da parte di altri peer. Ogni peer che riceve dati provvede a rendere disponibili gli stessi dati freschi verso gli altri peer con cui è in contatto. Un meccanismo di controllo della congestione permette ai peer padri di capire la velocità e i dati che deve, e può inviare ai figli.

Nemico della qualità di invio di dati nella topologia a rete mesh è il cosiddetto fenomeno del BottleNeck (collo di bottiglia), che si può presentare sotto due diverse forme: il *Bandwith Bottleneck* e il *Content Bottleneck* (12). Il primo tipo di ingorgo si crea quando la larghezza di banda disponibile aggregata da tutti i genitori ad un dato peer non è sufficiente grande rispetto la larghezza di banda del nodo, mentre il secondo tipo di ingorgo si ha quando la dimensione fisica di alcuni genitori di un determinato peer non è sufficientemente ampia da contenere il

contenuto utile del nodo. Per evitare tali problemi si assume che ogni genitore manda i pacchetti ai figli rispettando un meccanismo controllato della congestione. Ovviamente la larghezza di banda di ogni peer dipende solo dalle proprietà della struttura a overlay, ovvero larghezza di banda in entrata e uscita e grado di partecipazione dei peers.

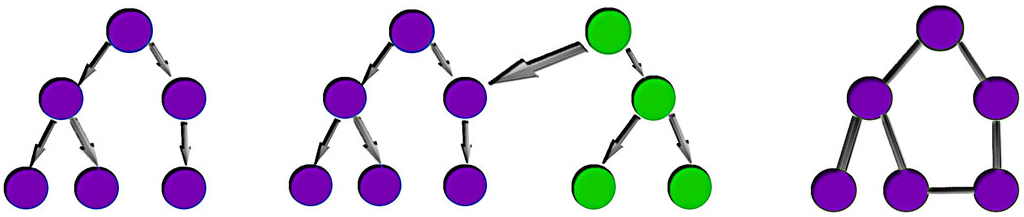


Figura 1.4: Architetture per reti strutturate.

1.3.2. Rete non strutturata

Algoritmo basato sul Gossip

Gossip è un protocollo/algoritmo epidemico che utilizza la randomizzazione per la diffusione di messaggi in un gruppo (13). Tale protocollo permettere ai peer dell'overlay network di inviare in flooding[15] a tutti i nodi della rete alcuni messaggi. Rientrano in questa categoria le informazioni di presenza che i nodi mandano periodicamente al resto della rete in modo che ogni nodo abbia una lista dei partecipanti (member table) da cui poter selezionare come sottoinsieme i suoi partner.

[15] Flooding: è un protocollo di instradamento usato dai router che inoltrano un pacchetto in ingresso su tutte le linee ad eccezione di quella da cui proviene. Ogni pacchetto in arrivo viene inoltrato su ogni linea di uscita eccetto quella da cui è arrivato. Questo algoritmo genera un vasto numero di pacchetti duplicati; in effetti, un numero infinito, a meno di non prendere qualche misura per fermare il processo.

Nell'algoritmo gossip un messaggio viene distribuito come segue: quando un nodo genera un messaggio, questo lo invia verso un sottoinsieme casuale di nodi suoi vicini. Quando un nodo riceve un messaggio per la prima volta ripete la stessa procedura appena descritta.

Negli ultimi anni gli algoritmi basati sul Gossip hanno guadagnato il riconoscimento di metodologie adatte alla progettazione di schemi di comunicazione robusti e scalabili. La scalabilità deriva dal fatto che questa tipologia di protocolli non richiede una sincronizzazione come quella per esempio che avviene nei tradizionali protocolli di multicast. Nei protocolli non deterministici come il Gossip la dinamica dell'esecuzione non è determinabile a priori. Tutti i protocolli non deterministici, sono in genere statici, ovvero o eseguono immediatamente la ritrasmissione oppure interrompono il flooding: infatti non necessitano di alcun intervallo di attesa per la ritrasmissione e quindi hanno una latenza minore dei protocolli deterministici.

Essi realizzano una sorta di meccanismo di replicazione molto utile per ovviare ad eventuali guasti di stazioni della rete o dei link che le collegano, poiché un nodo riceve una copia multipla del messaggio da parte di più stazioni. Nessun nodo all'interno della rete ha un ruolo specifico e quindi il danneggiamento di una stazione non impedisce agli altri di continuare a trasmettere messaggi. Inoltre non è affatto difficile aggiungere o rimuovere nuove stazioni all'interno della rete.

I protocolli di Gossip hanno due caratteristiche principali: scalabilità, ed un degrado della rete molto lento. Per quanto riguarda la scalabilità possiamo dire che le performance di questo tipo di protocolli non diminuiscono rapidamente quanto la crescita del numero di nodi del sistema ed inoltre ogni singola

stazione trasmette un numero fisso di pacchetti indipendentemente dal numero dei dispositivi presenti all'interno della rete.

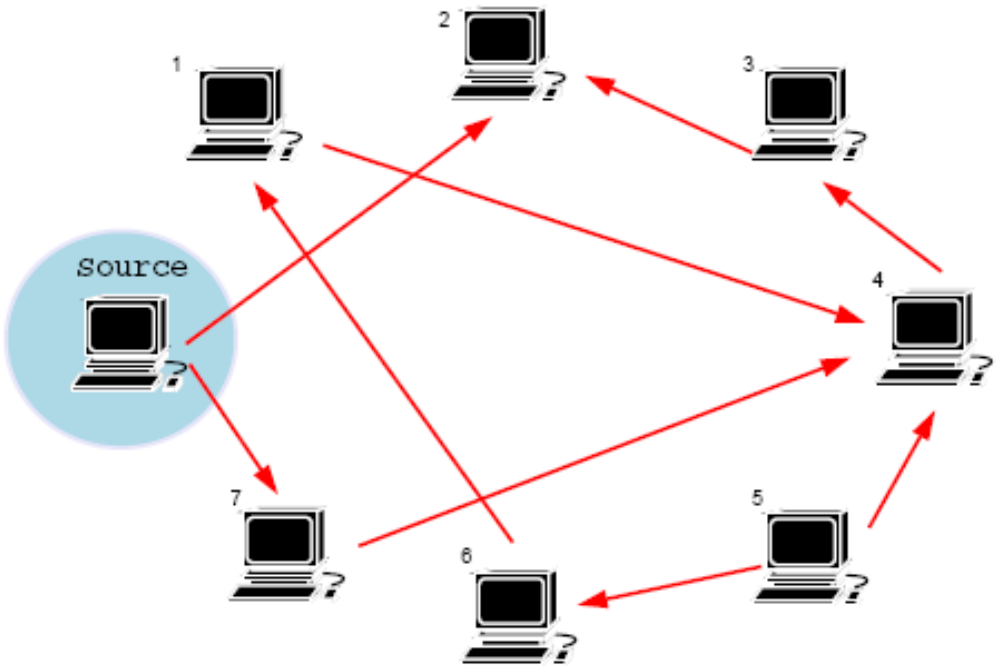


Figura1.5: Comunicazione tra peer mediante Algoritmo basato sul Gossip

Capitolo 2

La qualità del servizio nello streaming p2p

Il secondo capitolo affronta il tema della Qualità del Servizio dei sistemi p2p utilizzati per il live streaming. Il primo paragrafo introduce la tematica della qualità e i parametri metrici riguardanti il mondo p2p in genere; il secondo paragrafo si sofferma invece sugli aspetti che riguardano le applicazioni come le P2PTV che dunque forniscono contenuto multimediale in streaming; il terzo ed ultimo paragrafo tratterà quelli che possono essere gli aspetti sui quali intervenire per migliorare la qualità del servizio di un sistema p2p Live Streaming.

2.1. Verso la qualità del servizio

Il termine "Qualità del servizio" o più semplicemente QoS (dall'inglese Quality of Service) è usato per indicare i parametri usati per caratterizzare la qualità del servizio offerto dalla rete oppure gli strumenti e le tecniche per ottenere una qualità di servizio desiderata (14).

Quando è stata creata la rete Internet, non era stata percepita la necessità di QoS per le applicazioni. Infatti l'intera Internet segue la filosofia del ***best effort***, cioè il sistema garantisce di fare tutto il possibile per portare a termine un'operazione, ma non garantisce affatto che l'operazione verrà compiuta, né in che modo. Anche se il protocollo IP prevede 4 bit per il tipo di servizio (*type of service*) e 3 per la *precedenza* di ciascun pacchetto, questi bit sono largamente inutilizzati. Al crescere del numero e delle tipologie di servizi e del traffico offerto rispetto alle capacità della rete il problema della qualità del servizio ha cominciato a divenire importante e sempre più considerato.

Ci sono fondamentalmente due modi per fornire garanzie sulla Qualità del servizio:

- *Sovradimensionamento*: consiste nel fornire risorse di rete (di trasmissione, memorizzazione ed elaborazione) in abbondanza, abbastanza da soddisfare la domanda di picco attesa, con un sostanziale margine di sicurezza. Una soluzione semplice, ma alcuni credono che in pratica sia troppo costosa e non sia applicabile se la domanda di picco cresce più velocemente di quando predetto: disporre nuove risorse richiede infatti sempre tempo.
- *Priorità*: consiste nell'amministrare la banda disponibile, facendo in modo che i pacchetti a cui deve essere garantita una certa QoS ricevano in particolar modo un trattamento privilegiato. Per ottenere questo, bisogna risolvere due problemi:
    1) Identificare i pacchetti che devono ricevere un trattamento privilegiato

(*classificazione* o *discriminazione del traffico*). Si utilizzano due tipi di metodi:

- *Integrated services*, basato sulle prenotazioni: prima di iniziare una sessione che ha dei requisiti di QoS, l'applicazione deve "chiedere" alla rete se questa può garantire le prestazioni necessarie (*admission control*): la rete valuta se dispone delle risorse adeguate e in caso positivo accetta la prenotazione concedendo il servizio richiesto.
- *Differentiated services*, prevede che gli utenti della rete stipulino a priori un contratto che definisca la quantità massima di traffico "privilegiato" che essi possono generare e marchino tale traffico utilizzando il campo *Type of Service* (*TOS*) dell'header IP. In questo caso quindi le prenotazioni sono rigidamente "statiche".

2) Applicare a questi pacchetti identificati una disciplina di coda (*queue discipline*) che stabilisce in quale ordine verranno prelevati i pacchetti dalle varie code (es. Weighted Round Robin, Hierarchical Packet Fair Queuing, Hierarchical Fair Service Curve (15)).

In una rete a commutazione di pacchetto (PBN, *Packet Based Network*), come è la rete peer-to-peer, l'informazione da trasmettere viene suddivisa in pacchetti di dimensione abbastanza piccola; ad ognuno di essi viene aggiunta un'intestazione (header) che precede il contenuto informativo vero e proprio e contiene tutta l'informazione necessaria

affinché il pacchetto possa essere identificato e inoltrato fino alla sua destinazione finale, ovvero l'indirizzo del destinatario, la sua posizione (numero di pacchetto) all'interno del flusso dell'informazione che viene trasferito più tutta una serie di informazioni aggiuntive come ad esempio il livello di priorità del singolo pacchetto, la lunghezza del campo informativo, il tipo di informazione trasportata (per distinguere per esempio i pacchetti che trasportano contenuto informativo dai pacchetti che trasportano informazioni di protocollo o di altro tipo). I pacchetti vengono inviati individualmente attraverso la rete e vengono poi riassemblati, grazie al numero di pacchetto, nell'ordine originale quando arrivano sul nodo destinatario.

Dunque un pacchetto che attraversa una rete subisce un ritardo, legato a quattro componenti fondamentali, non tutte deterministiche:

- **Tempo di elaborazione**: è il tempo necessario a ciascun nodo per processare il pacchetto e deciderne l'interfaccia di uscita. Se un nodo agisce a diversi livelli di pacchetto, ciascun livello aggiungerà una sua componente di ritardo.
- **Tempo di trasmissione**: è il tempo necessario per trasmettere il pacchetto alla velocità della linea di trasmissione. È dato dal rapporto tra lunghezza in byte del pacchetto e velocità o capacità di linea.
- **Tempo di latenza di accodamento** (*queuing delay*): è dovuto al fatto che i pacchetti in uscita o in entrata non sempre vengono trasmessi/ricevuti immediatamente. Infatti la linea in uscita può essere occupata da altri pacchetti in corso di trasmissione/ricezione. In questo caso, il pacchetto viene salvato in una memoria temporanea del commutatore (buffer) detta *coda*, per venir trasmesso appena possibile. Il tempo atteso dal pacchetto

nella coda è detto *ritardo di latenza*, ma di esso non è possibile stabilirne a priori la durata in quanto varia in funzione della dimensione della coda.

- **Tempo di propagazione**: è il tempo necessario al segnale fisico per propagarsi lungo la linea di trasmissione fino al nodo successivo e da qui alla destinazione finale.

Legate alla questione del ritardo sorgono quindi due problematiche. La prima consiste nel fatto che alcune forme di comunicazione possano espressamente richiedere che il ritardo complessivo di trasferimento dei pacchetti rimanga strettamente entro un limite massimo, pena effetti indesiderati se non distruttivi della comunicazione stessa.

La seconda problematiche invece è la cosiddetta varianza massima del tempo di trasmissione (*jitter*), ossia il fatto che per un certo pacchetto il ritardo potrebbe essere diverso rispetto ad un altro pacchetto appartenente alla stessa comunicazione e ciò potrebbe avere una ripercussione negativa nel nodo destinazione per la ricostruzione dell'integrità dell'informazione. Infatti nel caso del live streaming una variazione eccessiva può comportare distorsioni o perdite temporanee non accettabili e si potrebbero manifestare forme di fermo immagine temporanee o in immagini di qualità inferiore.

Esistono diversi meccanismi per cercare di attenuare o rendere controllabili gli effetti del ritardo. Per esempio, per ridurre o limitare il tempo di trasmissione, si può ricorrere a tecniche di tipo *store and forward*, per cui all'interno del nodo si attende il completamento della ricezione e dell'elaborazione del pacchetto prima di inviarlo alla coda di trasmissione.

2.2. QoS nelle P2PTV

Abbiamo notato come il fattore tempo e le problematiche relative al ritardo che questo introduce siano di primaria importanza se si vuole garantire una QoS soddisfacente. Vogliamo ora estendere il discorso ai sistemi p2p progettati per il live streaming: è evidente che per questi tipi di architetture l'obiettivo principale è quello di aumentare il numero di peer che si connettono alla rete per usufruire di un certo servizio. Quindi un primo aspetto importante è dato dal fatto che il sistema ha l'esigenza di supportare un gran numero di peer visto e infatti i programmi per lo streaming offrono servizi a migliaia e talvolta milioni di utenti dando massima importanza alla garanzia di una buona QoS a tutti i nodi connessi alla rete. Un secondo aspetto di rilievo è che qualora il sistema non fosse in grado di garantire una sufficiente qualità del servizio gli utenti insoddisfatti potrebbero decidere di abbandonare la rete.

Tale fenomeno, caratterizzato dal comportamento dinamico degli utenti che accedono alla rete e/o la abbandonano, è conosciuto col nome di *Node Churn* e contribuisce anch'esso al cambiamento dello stato della rete.

Un'altra importante considerazione riguardo al QoS di un sistema p2p live streaming è il tempo di start necessario al canale (*Channel Startup Time*), ovvero il tempo necessario al player video per incominciare la visualizzazione dei contenuti multimediali presenti sul buffer della memoria. In sostanza si tratta del ritardo complessivo che intercorre tra la rete p2p e i peer ad essa connessi. Il valore di tale ritardo può dipendere dalle condizioni differenti di network, dalle dinamiche degli algoritmi di connessione dei peer e dal fatto che la maggior

parte degli utenti non ha un indirizzo IP diretto, bensì mascherato da un router NAT[16] o un firewall.

Un penultimo aspetto riguardante il QoS è la *qualità dello stream video*: il sistema p2p al fine di rendere disponibile lo stream di un video, permette il download dei blocchi di stream ai vari peer ad esso associati. La qualità dello stream video è misurata come il rapporto tra il numero dei blocchi a disposizione del client e il numero dei blocchi necessari per una perfetta codifica del video.

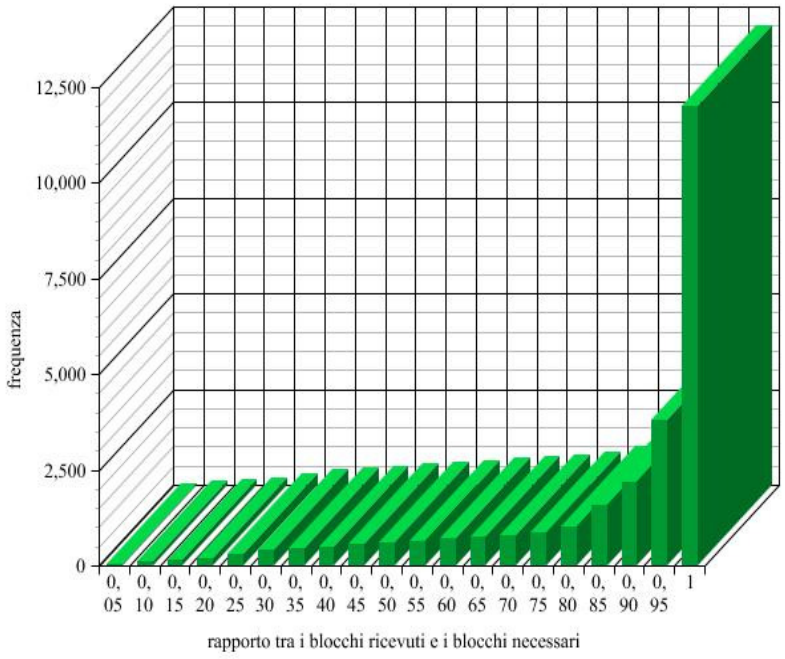


Figura 2.1: Grafico relativo al rapporto fra blocchi ricevuti e blocchi necessari

[16] NAT: Network Address Translation, ovvero traduzione degli indirizzi di rete, è una tecnica che consiste nel modificare gli indirizzi IP dei pacchetti in transito su un sistema che agisce da router all'interno di una comunicazione tra due o più host. Le connessioni effettuate da uno o più computer vengono alterate in modo da presentare verso l'esterno uno o più indirizzi IP diversi da quelli originali. Quindi chi riceve le connessioni le vede provenire da un indirizzo diverso da quello utilizzato da chi effettivamente le genera.

L'ultimo aspetto da considerare fa riferimento alla *correzione degli errori*. A causa della non predicibile congestione del traffico Internet, nel flusso dei pacchetti una frazione dei pacchetti stessi potrebbe andare persa. Se questa frazione diventa troppo grande, la qualità audio/video che l'utente percepisce diventa inaccettabile. A questo scopo, molti sistemi di streaming tentano di recuperare queste perdite sia ricostruendo i pacchetti persi attraverso una ritrasmissione degli stessi (trasmissione ridondante), sia attraverso una richiesta di trasmissione esplicita da parte del nodo client, oppure mascherando le perdite con l'interpolazione dei dati mancanti con quelli ricevuti.

2.3. Sviluppi futuri per il miglioramento del QoS

Il mercato non ha ancora favorito la nascita di servizi QoS end-to-end, ovvero in grado di garantire vincoli sulla QoS di un flusso di dati scambiati tra utenti remoti. Alcuni credono che una rete *stupida* cioè sovradimensionata, che offra cioè sufficiente banda per la maggior parte delle applicazioni e per la maggior parte del tempo, sia già economicamente la migliore soluzione possibile, mostrando poco interesse a supportare applicazioni non-standard capaci di QoS.
La rete Internet ha già accordi complessi tra i provider e sembra che ci sia poco entusiasmo nel supportare il QoS attraverso connessioni che interessano reti appartenenti a provider diversi, o sugli accordi circa le politiche che dovrebbero essere sostenute al fine di poterle supportare.

Gli scettici sul QoS indicano che se si scartano troppi pacchetti su una connessione elastica[17] a basso QoS, si è già pericolosamente vicini al punto di una congestione per le applicazioni inelastiche[18] ad elevato QoS, non essendoci più modo di scartare ulteriori pacchetti senza violare i contratti sul traffico (16).

[17] connessioni elastiche: connessioni utilizzate da applicazioni che possono funzionare anche su reti con prestazioni molto degradate e usare tutta la banda a disposizione anche se questa è abbondante.
[18] applicazioni inelastiche: applicazioni che richiedono un certo livello di banda per funzionare, se ne ottengono di più non la sfruttano e se ne ottengono di meno non funzionano affatto. Sono queste applicazioni che rendono
necessaria l'adozione di misure per garantire una certa QoS.

Capitolo 3

La performance nello streaming p2p

Il terzo capitolo affronta il tema della performance dei sistemi p2p, il primo paragrafo descriverà quali sono i criteri che valutano la performance sotto quattro diversi punti di vista: livello utente, risorse, overlay di rete e livello IP. Il secondo paragrafo si sofferma sull'importanza che assume il principio di Equità (fairness) nello studio delle prestazioni di un sistema streaming p2p, mentre nell'ultimo paragrafo verrà trattata l'identificazione dei nodi stabili che rendono il servizio più affidabile e offerte alcune soluzioni in vista di possibili perdite di informazioni durante la trasmissione di contenuto informativo tra peers comunicanti.

3.1. Criteri di misurazione della performance

In questa sezione discutiamo una serie di questioni rilevanti e i problemi che svolgono un ruolo importante nella valutazione della performance delle applicazioni P2P di file sharing. La discussione si basa sulla classificazione delle questioni rilevanti rispetto a quattro diversi punti di vista (17):

- utente;
- risorse (livello applicativo);
- overlay di rete (livello applicativo);
- livello di rete.

La figura 3.1 mostra tale classificazione e verrà nei prossimi paragrafi discussa punto per punto.

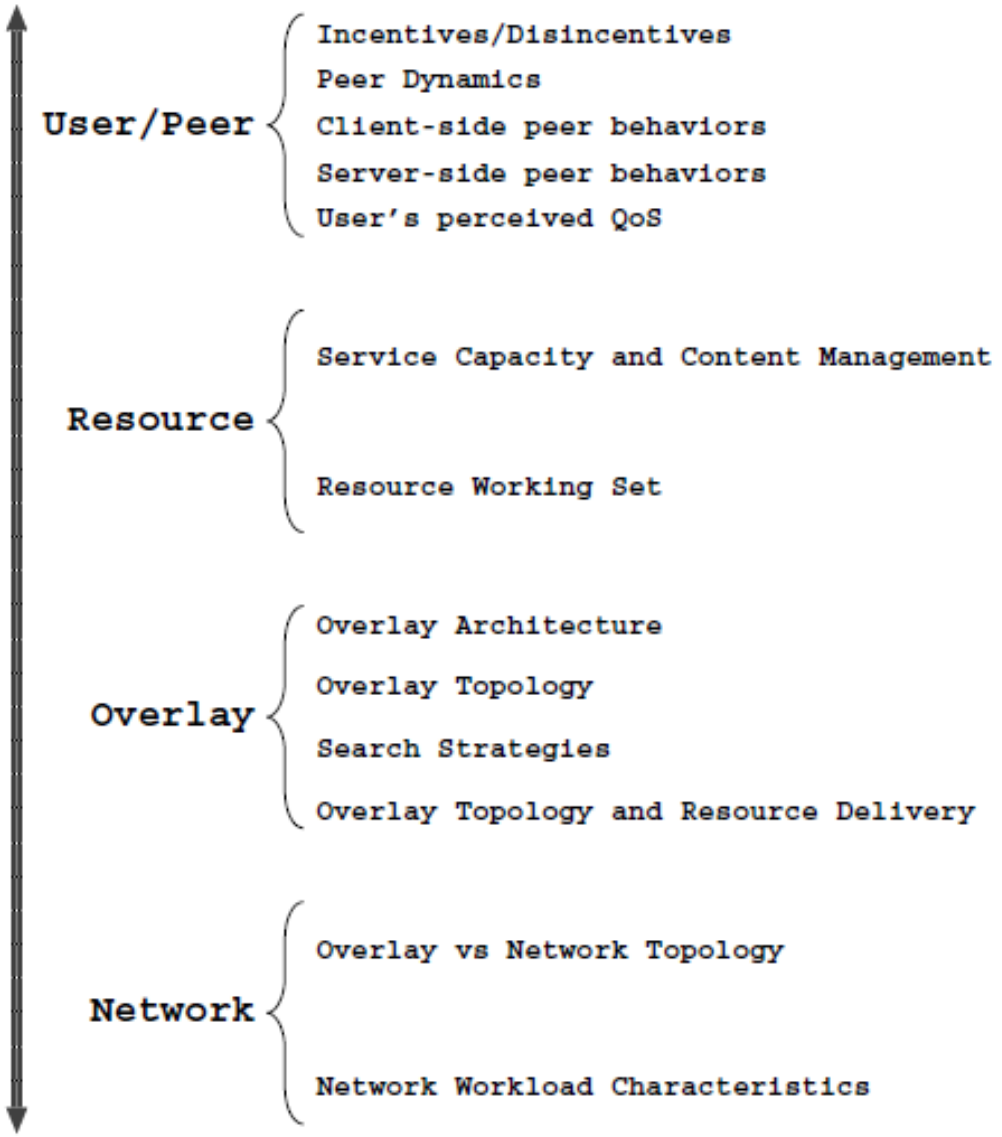


Figura 3.1: Classificazione dei parametri che influenzano la performance nel P2P

Punto di vista dell'utente

Questo livello di classificazione comprende tutti i problemi di prestazioni che sono collegati al comportamento dell'utente/peer. Le domande che un utente di un'applicazione basata sul peer-to-Peer si pone sono essenzialmente due: qual è la probabilità che ho di individuare la risorsa che richiedo? Quanto tempo occorre per trasferire tale risorsa al mio computer? Per rispondere a questo tipo di domande si possono individuare i seguenti aspetti fondamentali.

• Il primo è la cooperazione: senza la cooperazione fra i nodi della rete l'utente potrebbe percepire lunghi ritardi di trasferimento e talvolta potrebbero essere non in grado di ricevere/trasferire la risorsa. Inoltre, la cooperazione è essenziale per partecipare alla ricerca di una determinata risorsa e quindi ha un impatto sulla probabilità di localizzare una risorsa. Purtroppo gli utenti hanno dei disincentivi naturali nella collaborazione: gli utenti spesso possiedono limitate risorse di banda e di elaborazione e consentendo un buon upload agli altri peers finirebbero per rischiare di apportare un ritardo ai propri download. Ciò conduce al cosiddetto problema del "*free-riding*", la cui conseguenza è che la maggior parte delle richieste di risorse sono dirette verso un numero limitato di peers, che sono disposti a condividere le proprie risorse. Si verifica quindi la "Tragedy of commons", in cui i partecipanti tendono egoisticamente a massimizzare la propria attività, a scapito di quella complessiva del sistema. In questo contesto, la progettazione di strategie di incentivi differenti è un problema cruciale: si potrebbe infatti decidere un tempo medio di download per ogni peer favorendo maggiore cooperazione.

• Secondo aspetto di interesse per l'utente dipende dalla dinamicità del peer. In una rete di P2P gli utenti possono accedere e abbandonare il sistema in qualsiasi momento, introducendo fluttuazioni sul numero di peers attivi in un certo istante di tempo. Questa incertezza può introdurre ritardi nel trasferimento delle risorse e nella capacità di individuare una particolare risorsa. L'abbandono del sistema può essere dovuto all'impazienza utente oppure a causa di risorse limitate (18).

Un altro fattore da cui può dipendere il trasferimento di una risorsa è il comportamento lato client degli utenti. Quando un peer esegue una ricerca su una risorsa, l'applicazione fornisce all'utente un elenco di peer in possesso di una copia della risorsa richiesta. Spesso vengono incluse anche informazioni aggiuntive sui peers che la possiedono, come ad esempio l'ampiezza di banda tra il peer e il suo ISP, il numero di altri peer che stanno scaricando (o in alcuni casi che sono in coda) da questo nodo, la qualità della risorsa, ecc (19). Tutte queste informazioni possono essere utilizzate da un utente per selezionare la copia più opportuna della risorsa richiesta.

• Un ultimo aspetto suggerisce che il comportamento di un peer lato server gioca un ruolo importante nella QoS percepita dall'utente. Infatti il peer che funge da server può utilizzare strategie diverse per condividere le proprie risorse tra l'insieme dei nodi che stanno scaricando da esso o tentano. Lo scopo di queste strategie può essere incentrato sull'evitare lunghi ritardi nel download oppure sull'incentivazione attuando meccanismi di differenziazione del servizio. In questo contesto, il disegno delle politiche di servizio devono essere attentamente sostenuta dalla valutazione del loro impatto sulla QoS percepita dall'utente e sulle prestazioni del sistema complessivo.

Punto di vista delle risorse

Osservando un'applicazione basata sul peer-to-peer dal punto di vista delle risorse coinvolte, un indice di prestazione rilevante è la capacità di servizio del sistema.

Se un peer ha una risorsa popolare e molti altri peers lo richiedono, i peer richiedenti possono andare incontro a scarse prestazioni. Tuttavia, non appena uno dei peer ha completato il download della risorsa (o in qualche caso, non appena si completa il download di una frazione di essa) il sistema avrà due server per gestire le richieste dagli altri peer. In questo modo si evolve il processo di diffusione, moltiplicando il numero di server per la risorsa e aumentando lo througput che il sistema può offrire ai peers (20). Questa metrica è detta "*aggregate upload service capacity*" e calcola l'upload di banda per i seed, nodi che possiedono l'intero file, e per i peers che devono ancora terminare il download.

Esiste tuttavia un'altra metrica riferita alle risorse ed è la cosiddetta "*per peer download througput*", che in particolare rappresenta il througput medio di download ottenuto per ogni nodo della rete. La capacità di un sistema P2P di fornire un servizio (service capacity) in sostanza è il numero di copie di ciascuna risorsa che la rete può fornire a chi la richiede, dunque maggiore è il numero di repliche, maggiore è la capacità. Aggiungiamo che è una metrica di tipo dinamico perchè è in continua variazione, dal momento che i peers possono accedere e abbandonare la rete in ogni istante. Possiamo analizzare le prestazioni del sistema in due differenti fasi: quella in *regine transitorio* e quella in *regime stazionario*. La fase in transitorio valuta la performance dell'applicazione P2P quando un nodo esegue una richiesta di un file molto popolare che è appena stato

introdotto all'interno della rete. All'inizio ci saranno solo pochi peers ad avere una copia del file che possono offrire ad altri nodi, e col passare del tempo all'aumentare della popolarità del documento seguirà una crescita esponenziale della capacità del servizio. Poi le richieste del file stesso diminuiscono (fase stazionaria) fino a stabilizzare lo througput dei singoli peer.

Dunque una questione importante è insita anche nella caratterizzazione delle risorse condivise, in termini di numero, popolarità e dimensione (21).

Punto di vista dell'overlay di rete

L'architettura P2P costruire un overlay (virtuale) di rete al di sopra della rete fisica, che viene gestito a livello di applicazione e che caratterizza la topologia della rete.

La ricerca di una risorsa e la sua consegna ad un determinato peer che ne esegue una richiesta sono fortemente influenzati da due proprietà: l'architettura della struttura a sovrapposizione (*overlay network architectures*) e la topologia della struttura a sovrapposizione (*overlay network topology*).

• La prima rappresenta il modo in cui i nodi dell'applicazione si auto-organizzano per creare un'infrastruttura da utilizzare per la ricerca e la distribuzione di risorse. Per esempio esistono architetture centralizzate, distribuite ma non strutturate, distribuite gerarchicamente, distribuite mediante tabelle hash, ibrideP2P (22).

• La seconda invece rappresenta le proprietà di connettività proprie dei peers che compongono la rete. La valutazione delle prestazioni in termini di algoritmi di ricerca su una topologia di

rete overlay può essere valutata in termini di numero di messaggi originati da una query, oppure in base al tempo di ricerca, o alla probabilità di successo della ricerca stessa. La scelta di una idonea topologia per un sistema peer-to-peer inoltre influisce anche sul processo di erogazione delle risorse.

Quando per esempio è possibile identificare un'insieme di utenti che generano richieste di risorse condivise, la performance complessiva del sistema può migliorare notevolmente se si forza dall'esterno la costruzione di una topologia di consegna ad overlay ad hoc per quella determinata rete.

Punto di vista del livello di rete

Oggigiorno le applicazioni basate sul P2P stanno ricavandosi uno spazio sempre più grande rispetto all'intero traffico della rete Internet: recenti studi sottolineano che l'attività di file sharing sta cominciando a dominare l'ampiezza di banda complessiva in certi segmenti di Internet (oltre l'80% secondo questo studio (23)).

Inoltre, a causa della mancata corrispondenza tra overlay e topologie di rete, grandi quantità di traffico P2P vengono inserite in collegamenti di rete più costosi. Da queste motivazioni appare chiaro l'importanza della caratterizzazione del traffico P2P, la valutazione del suo impatto sulla rete IP sottostante, e la valutazione delle strategie per la gestione del traffico.

3.2. Il principio di equità

Senza dubbio il principio cardine che sta alla base dei sistemi p2p, e quindi vale anche per il live streaming delle applicazioni P2PTV, è il principio di Equità (o *fairness*).

Il significato della parola equità deriva dall'idea che maggiori sono i nodi che condividono contenuto informativo e maggiore sarà il corrispettivo rate di download. Il tutto quindi per cercare di favorire lo througput di ciascun nodo della rete. In presenza di sistemi in cui il numero di partecipanti è considerevolmente grande, come ad esempio nei sistemi che stiamo trattando in questo progetto, risulta particolarmente difficile riuscire ad attribuire ad ogni peer un eguale limite di download.

Lo scenario che si crea quando un nodo fa il suo primo ingresso nella rete è che non si sa a priori se tale nodo nel passato abbia già partecipato alla condivisioni di file. Sarebbe dunque opportuno al fine di migliorare le prestazioni di tutto il sistema attribuire al suddetto peer il massimo rate per il download nella fase di regime transitorio, ovvero nei primi istanti dopo l'accesso, in quanto gli si da così la possibilità di incominciare da subito a condividere in breve tempo i file con gli altri nodi, dopodiché trascorso un certo lasso di tempo, entrando nel regime stazionario, gioca il suo ruolo il principio di equità assegnando un idoneo valore al rate download per il nodo in esame in maniera proporzionale a quanto il nodo a partecipato alla condivisione.

Per scoraggiare il comportamento da "free riders" degli utenti della rete i quali beneficiano di un servizio non curandosi di condividere a loro volta le risorse che hanno a disposizioni, alcune reti P2P hanno adottato un sistema di assegnazione di

crediti, in cui si prevede un premio ai nodi che condividono maggiori quantità di informazioni offrendo loro un tasso di servizio più rapido ed efficiente (24).

3.3. Nodi stabili

I nodi stabili sono un sottoinsieme dei nodi che compongono l'intera rete peer-to-peer. Essi sono infatti quei nodi che restano sempre connessi e vengono in contro notevolmente ai Source Provider (i detentori della sorgente) occupandosi anchessi della distribuzione del contenuto in streaming verso altri nodi, i quali invece hanno maggiore facilità di abbandono della rete quando credono.

In questo contesto si parla di Indice di Stabilità (*Stability Index*) in quanto è utile indagare quale sia il grado di stabilità di un nodo, indicando con un valore vicino a 0 se si tratta di peer instabile, o vicino a 1 se invece è prossimo alla stabilità. Il calcolo di tale indice evidenzia che se col passare del tempo un nodo rimane stabile all'interno della rete, allora esso tenderà a rimanerlo anche in futuro. Se poniamo una certo valore di soglia, sarà possibile anche stabilire quali saranno i peer che superato tale valore possono passare dall'essere instabili all'essere stabili.

Questa tecnica però sottende due svantaggi: qualora un numero alto di peer facesse ingresso nella rete contemporaneamente e si creerebbe il problema derivato dal considerare tutti questi nodi stabili simultaneamente. Un altro problema si evidenzia quando all'inizio della sessione, non vi è ancora presenza di nodi stabili e si impiegherà del tempo prima di riuscire a costruire per intero la rete a overlay come sovrapposizione di più livelli.

È possibile risolvere queste dinamiche mediante l'utilizzo di efficienti algoritmi di predizione di stabilità, oppure l'inserimento di un nuovo nodo che si trova al primo livello, chiamato Labeled Tree (*LB Tree*), che permette il recupero di eventuali dati persi e offre migliori servizi ai peer di secondo livello.

Un Labeled Tree assegna ad ogni nodo un'etichetta che indica la sua posizione all'interno del primo livello. La strategia utilizzata è basata sul considerare in un primo momento tutti i nodi eccetto la sorgente (nodo root) come peers di secondo livello, e quindi non stabili. Col passare dei secondi i nodi assumono la connotazione di stabilità e possono ora entrare a fare parte dell'albero di primo livello. La relazione tra l'indice di stabilità e il Labeled Tree è la seguente: tanto maggiore è la vicinanza dalla root, tanto più alto è l'S-Index, e tale indice sappiamo crescere in relazione al trascorrere del tempo.

Dunque tale relazione risulta essere persistente durante tutta la durata della sessione di download. Per quanto riguarda il primo livello della sovrastruttura occorre fare una precisazione: non soltanto i nodi facenti parte di questo livello vengono definiti stabili, ma anche le connessioni che questi possono instaurare con nodi del livello sottostante possono considerarsi stabili e garantire basse possibilità di perdita di dati.

La perdita di contenuto informativo nel periodo di trasferimento di un file è un problema di primaria importanza. Per ridurre questo tipo di inconveniente e mantenere soddisfacenti performance nel sistema P2P una soluzione sarebbe utilizzare i cosiddetti *side link* tra le coppie di nodi che scambiano dati. Questo tipo di collegamento viene utilizzato infatti per eseguire richieste di recupero dati e conseguenti risposte utilizzando un residuo della banda a disposizione del

peer. Ogni nodo può stabilire molteplici side link, uno per ogni blocco perso, ottemperando a due specifici criteri: un side link uscente da un nodo dovrebbe essere connesso ad un nodo molto vicino alla sorgente della rete; e i due peers coinvolti nel side link è preferibile che non appartengano allo stesso sottoalbero altrimenti il recupero potrebbe non portare ad alcun risultato.

Capitolo 4

Caso di studio: SopCast P2PTV

In questo capitolo viene presentato uno dei principali esempi di applicazione P2PTV chiamata SopCast. Prima di addentrarci nell'analisi dell'architettura software verrà fatta una breve introduzione circa le Peer-To-Peer Television, evidenziandone peculiarità, caratteristiche strutturali e modelli di riferimento. Verrà trattato il protocollo BitTorrent sul quale, secondo molti ricercatori, SopCast sia fortemente improntato e successivamente elencate le principali client applications per trasmissione di canali TV su reti p2p. Nel secondo paragrafo ampio spazio a SopCast, sviluppo e caratterizzazione. Seguirà poi un paragrafo in cui si citano gli esperimenti condotti da un team di ricercatori che ha analizzato il protocollo nello specifico, identificando le tipologie di pacchetti che viaggiano tra i nodi connessi e la struttura a tre livelli di sovrapposizione. Nell'ultimo paragrafo viene indagata la natura dinamica del sistema SopCast, considerando anche le possibili implicazioni a livello topologico.

4.1. Introduzione alle P2PTV

Le P2PTV, ovvero Peer-To-Peer Television, si sono enormemente diffuse nel corso dell'ultimo lustro, soprattutto in Cina e Stati Uniti.

I contenuti dei flussi video riguardano tipicamente canali televisivi provenienti da tutto il mondo. La peculiarità di questa tecnologia è che basandosi sull'utilizzo di reti peer-to-peer non servono necessariamente server di elevate capacità per trasmettere i flussi video dei canali che i suddetti flussi vengono ritrasmessi dagli stessi utilizzatori della rete secondo il paradigma del peer-to-peer. Quindi non deve essere un server centrale a trasmettere tutti i flussi video, il server trasmette il flusso video a un certo numero di utenti (peer), questi a loro volta ritrasmettono il flusso video ad altri utenti (peer) generando una rete che può raggiungere anche milioni di utilizzatori. Il programma installato dall'utente si occupa di rintracciare e scaricare, tra gli altri utenti, i pezzi che mancano e di inviare quelli in proprio possesso ad altri. Il video viene ovviamente riprodotto con alcuni secondi di ritardo per permettere al numero maggiore possibile di peer di scaricare tutti i pezzi del filmato.

Alcune delle più diffuse applicazioni di P2PTV che verranno trattate più a fondo nei paragrafi successivi sono SopCast, CoolStreaming, PPStream, TVants, PeerCast, Zatoo, Tribbler, LiveStation.

Nelle reti P2P come si può dedurre dal modello ISO/OSI gli utenti contribuiscono a formare un strato (overlay) a livello applicativo che si va a porre al di sopra ad uno strato sottostante a livello di rete (underlying layer). Questa situazione è graficata nella figura sottostante.

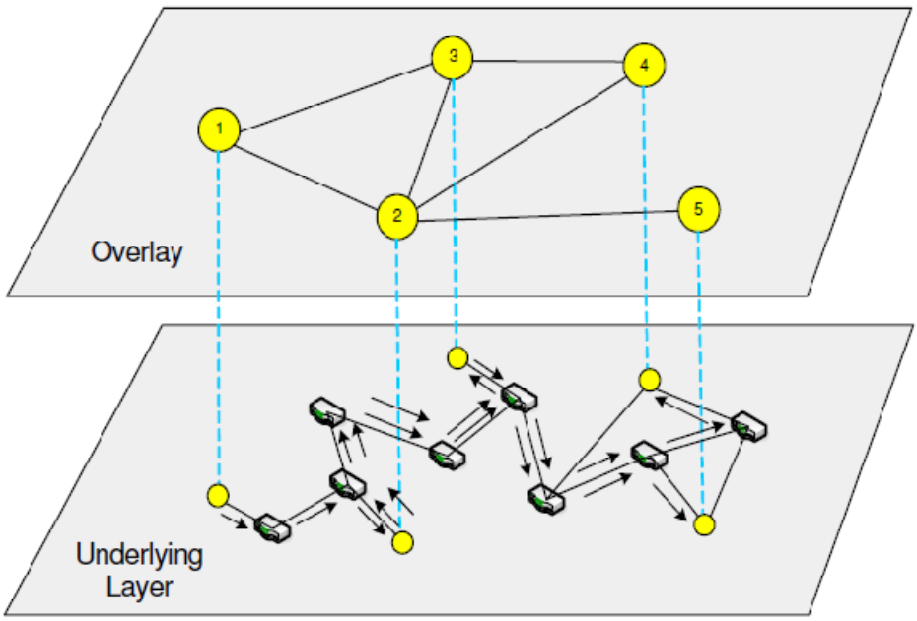


Figura 4.1: Struttura a due livelli di una rete P2P

Il modo più semplice per realizzare la trasmissione dei dati è deviare i percorsi (*routes*) sullo strato superiore senza considerare lo strato sottostante. Capire come funzioni questa sovrapposizione di livelli è un problema molto attraente. Esistono protocolli diversi che utilizzano meccanismi diversi di incentivazione e differenti algoritmi di selezione tra peers. Molti ricercatori stanno lavorando cercando di proporre algoritmi efficienti per migliorare le prestazioni delle applicazioni.

La maggior parte delle applicazioni P2PTV oggi sono proprietarie percui il codice sorgente non viene reso disponibile. Quindi l'unico modo per studiare la rete P2P è cercare di comprenderne il comportamento, la distribuzione del traffico, la topologia di overlay e l'algoritmo di selezione dei peer. Oltre a questo, si cerca anche di analizzare a fondo la

sovrapposizione dinamica del sistema P2P ponendo l'attenzione sull'impatto che possono avere i peer dinamici.

Nel primo capitolo si è detto che le principali architetture di riferimento per lo streaming sono: tree-based e mesh-based. Aggiungiamo ora che molti servizi di P2PTV usano l'architettura mesh-based, e il software che analizzeremo in questa tesi, SopCast, sfrutta proprio questo modello. Molti ricercatori sostengono che SopCast sia un sistema di P2PTV improntato su BitTorrent, mentre altri ritengono che SopCast utilizzi lo stesso algoritmo di BitTorrent solamente all'avvio della connessione, dopodiché preveda la presenza di un processo decentrato di scoperta peer anziché il Tracker come per BitTorrent.

4.1.1. BitTorrent

BitTorrent è uno dei più popolari protocolli P2P di file swarming[19] finalizzato alla distribuzione e condivisione di file nella rete, scritto in Python, fondato da Bram Cohen e sviluppato dalla BitTorrent Inc (25).

È un protocollo costruito su TCP/IP (Transmission Control Protocol/Internet Protocol) e HTTP (HyperText Transfer Protocol) o HTTPS (HTTP Secure) (26).

Il meccanismo di condivisione avviene nel seguente modo: i file originali che si vogliono condividere vengono spezzettati in tanti piccoli frammenti (detti *chunks*), di dimensioni pari a 256 KB che verranno inviati a chi li richiede, una volta che tutti i frammenti di un file vengono scaricati il file

[19] File swarming: con il termine "to swarm" (in italiano sciame) si indica la totalità dei seed (coloro che possiedono per intero il file in condivisione) e dei peer (coloro che stanno scaricando le parti del file) che condividono lo stesso file .torrent. In pratica, se un torrent è condiviso da 10 seed e altrettanti peer, lo swarm ad esso relativo sarà equivalente a 20 fonti.

viene ricomposto dal software client del destinatario. Il vantaggio è che in questo modo è possibile scaricare i frammenti di file che ci servono da più fonti, frammenti che a nostra volta siamo obbligati ad inviare a chi ne ha bisogno.

Le principali entità in gioco sono fondamentalmente cinque:

1. **I Peers (Leech)**: Sono coloro che stanno scaricando un file ma non lo hanno scaricato completamente, i peers condividono i frammenti di quel file che per il momento posseggono.

2. **I Seed**: Sono coloro che hanno diffuso il file originale o che sono riusciti a scaricarlo completamente, i seed sono molto importanti in quanto solo loro possono fornire informazioni su tutti i frammenti che compongono quel file e ovviamente solo loro hanno tutti i frammenti.

3. **Il Tracker**: Un server centrale che svolge il ruolo di "vigile urbano", il tracker infatti coordina le richieste degli utenti che stanno cercando di scaricare dei file e fornisce informazioni su chi ha i frammenti che servono, Il Tracker non contiene i file che vengono condivisi e i file non transitano attraverso di lui.

4. **I file .torrent**: Sono dei piccoli file con estensione .torrent che non contengono il file da scaricare ma solamente delle informazioni su di esso, come ad esempio il nome del file, la sua dimensione, tutti i frammenti nel quale è stato diviso il file originale (in sostanza l'hash di ciascun blocco in cui il file è stato suddiviso) oppure

l'indirizzo del Tracker da contattare identificandolo mediante URI (Uniform Resource Identifier). Per scaricare un qualsiasi file tramite BitTorrent è sempre necessario procurarsi questo file, per trovarli è necessario utilizzare un motore di ricerca specifico.

5. **Client BitTorrent**: è un programma installato nel pc dell'utente che permette di scaricare i file utilizzando il protocollo BitTorrent. Per iniziare il download di un file questo programma ha bisogno del suo file .torrent.

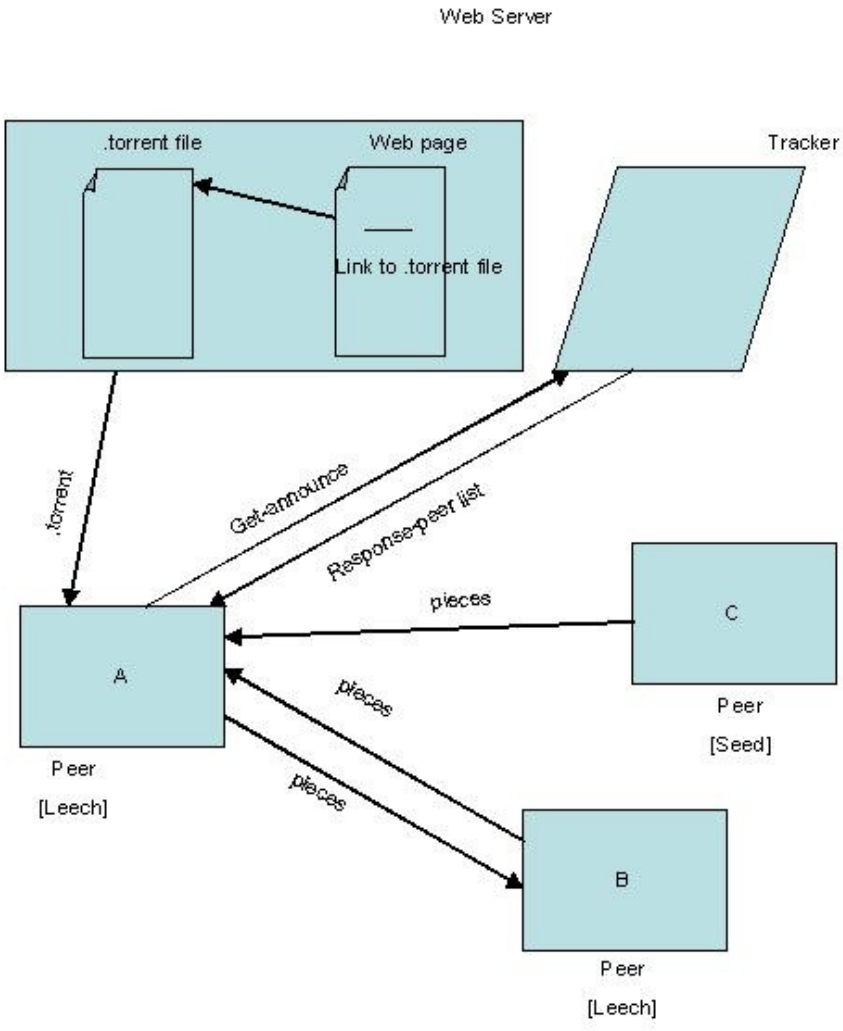


Figura 4.2: Comunicazione nel protocollo BitTorrent

Quando i peer vogliono scaricare un file, devono per prima cosa avere il relativo file .torrent. I peer quindi risolvendo il file .torrent, ottengono l'URI del tracker e lo contattano. Il tracker risponde alle richieste dei peer e provvede a fornire l'indirizzo IP degli altri peer inclusa la sorgente dei dati. Dopodichè il peer si connette agli altri peer mettendoli a conoscenza dei chunk che possiede e infine scarica dagli altri peer le parti mancanti del file che desidera ottenere. I peer hanno la necessità di decodificare il codice hash quando scaricano un chunk e lo confrontano con il file .torrent per verificare se hanno scaricato il chunk corretto.

Un aspetto importante da considerare ed anche uno scenario che spesso si riscontra in condizioni reali è che molti peer usano le risorse di rete ma subito dopo aver completato il download del file la lasciano, non permettendo più agli altri leech di ottenere le parti mancanti dal peer che abbandona.

Il *principio di Equità* (Fairness) di BitTorrent è garantito dal criterio di **tit-for-tat** che significa che un peer può partecipare attivamente al sistema solo se disposto a condividere il proprio contenuto con gli altri peer della rete, possibilmente in modo equo.

Tale criterio è realizzato dal meccanismo **"interest / chock / unchock"**. Ogni peer ha la responsabilità di aggiornare la propria velocità di download: i peer cercano di scaricare dal maggiore numero possibile di peer, e forniscono dati ad i peer che garantiscono una alta velocità di download, gli altri vengono scartati (choke). Questo significa che un peer fornisce servizi solo a coetanei che forniscono risorse ad esso e inoltre se un peer viene scartato diventa unchocked e

successivamente non sarà più bisogno costruire nuovamente un collegamento con esso.

C'è un altro meccanismo chiamato **optimistic unchoking**. Ogni peer BitTorrent ha un elenco di peer preferiti ai quali fornire risorse. In optimistic unchoking un peer prende un altro peer che non sia nella lista dei suoi preferiti e lo aggiunge ad essa. Se questo nuovo nodo possiede maggiore velocità di upload rispetto ad uno appartenente alla lista, allora esso andrà a sostituire nella lista in nodo a più minore velocità di upload. Quando un utente fa accesso alla rete, optimistic unchocking aiuta questi nuovi nodi. I peer con contenuti permetteranno ai coetanei che non condividono o ai nuovi peer di scaricare una parte del contenuto, il peer ricevente può condividere la stessa porzione di dati ricevuti e scambiarli con altri peer. I Peer che hanno utilizzato le risorse di altri peer, senza contribuire a fornire in cambio le proprie sono chiamati *"free rider"*. Per questo motivo nel sistema BitTorrent un peer scaricherà per primo il chunk[20] più raro onde evitare proprio l'abbandono della rete da parte del seeder detentore del pezzo di file.

Oltre ai vantaggi, BitTorrent ha anche dei limiti. Ad esempio, il tracker è un punto vulnerabile. È noto come il fallimento di un singolo peer non influenzerà la rete. Ma il fallimento del tracker rende impossibile ad un nuovo peer l'accesso alla rete.

[20] Chunk: Blocco di memoria residente nello heap. Ogni chunk è composto da un header (o intestazione) di dimensione costante 8 byte seguito da uno spazio di memoria variabile, la cui dimensione minima è la stessa utilizzata per l'intestazione e cresce seguendo l'ordine dei suoi multipli.

4.1.2. Applicazioni P2PTV disponibili

SopCast

SopCast è un software semplice e facile da usare. SopCast utilizza la tecnologia P2P per trasmettere i dati in modo molto efficiente. Ogni client può utilizzare un broadcast a basso costo per trasmettere il proprio programma. Il video broadcasting non ha bisogno di un super-server o di una grande larghezza di banda. I clienti riescono a costruire un buon sistema di rete live paragonabile a siti web commerciali su larga scala. SopCast è l'applicazione motivo di analisi nel proseguo della tesi e lo introdurremo dunque nel dettaglio nel successivo paragrafo (27).

CoolStreaming

CoolStreaming è una tecnologia di peer-to-peer TV che permette ai clienti di condividere dei contenuti televisivi con altri utenti. La tecnologia di base è simile a BitTorrent. CoolStreaming chiamato anche DONet, è una data-driven overlay network per lo streaming media. I vantaggi di questo software sono la possibilità di utilizzare al massimo la larghezza di banda e ottenere dunque migliori prestazioni. Ma questo programma è attualmente in fase di test periodo, quindi non è particolarmente stabile (28).

PPStream

PPStream è stato fondato da due ingegneri nella provincia della Repubblica Cinese di Sichuan ed è stato annunciato nel 2005. PPStream è stata la prima piattaforma al mondo di video su Internet ad integrare le tecnologie "P2P live" e "video-on-

demand". Il software client può essere utilizzato come una pagina web o come un programma desktop. PPStream è simile a BitTorrent e fornisce circa 90 canali. Si possono trasmettere programmi televisivi in modo stabile e senza problemi per gli utenti a banda larga. Nel 2008 PPStream è diventato la più grande piattaforma cinese a fornire servizi di network televisivo (29).

TVants

TVants è anch'esso un software basato su P2P TV. Un utente di TVAnt può impostare il proprio database di server tracker privato per raccogliere i diversi canali o condividere i propri file video in linea con chiunque sia possessore di una connessione a banda larga. Nel gennaio 2007 TVants collabora con Dream Windows per sviluppare un potente sistema di podcast, Vlog Ccants (30).

PeerCast

PeerCast è software open-source di radio P2P. E' stato annunciato nel corso del 2002 in Giappone. Confrontato con altri software P2P, i flussi di scaricamento PeerCast invece dei file, scambiano contenuti in tempo reale con altri clienti. Pertanto non necessita di memorizzare in locale i flussi di dati che arrivano ai peer collegati alla rete (31).

Zatoo

Zattoo è stato progettato in Svizzera. È un attuale riferimento ai canali europei e permette l'aggiunta di canali a pagamento. È stato infatti progettato per consentire solo a determinati utenti di guardare i programmi tv trasmessi. Al

momento il servizio è limitato alla Svizzera, Francia, Spagna, Germania, Norvegia, Danimarca e Regno Unito, ma prevede di estendere i servizi ad altre regioni europee. Lo svantaggio è che non è possibile utilizzare Zattoo sul sistema Linux (32).

Tribbler

Tribler è un software open-source P2P progettato dalla Delft University of Technology. Tribler è anche un'applicazione basata sul protocollo BitTorrent, ma ha aggiunto funzioni multiple principalmente focalizzate sull'interazione sociale e lo streaming video P2P. Alcune di queste particolarità riguardano le raccomandazioni, il friend-aided download e il video-on-demand. Inoltre è possibile utilizzare Tribbler sia sotto Linux che sotto Windows (33).

LiveStation

LiveStation è una piattaforma tecnologica basata su P2P che è sviluppata da Skinkers Ltd. Ha limite di banda, 256 Kbps minimi per upstream e 800kbps in downstream. Gli utenti di LiveStation possono aggiungere i vari canali alla propria directory e possono anche aggiungere numerose personalizzazioni circa la scelta dei contenuti video che preferiscono visualizzare (34).

4.2. Cos'è SopCast?

SopCast è un tipico esempio di sistema P2PTV proprietario: è un sistema di trasmissione live streaming basato sulla tecnologia P2P (35). Ed è una delle applicazioni P2PTV più popolari. In *Figura 4.3* viene presentata l'interfaccia utente

del client dell'applicazione SopCast e una lista di canali a cui l'utente può avere accesso, mentre in *Figura 4.4* viene visualizzata l'interfaccia del server provider. SopCast è stato sviluppato da un gruppo di studenti della Fudan University in Cina. Gli utenti possono usarlo per guardare la TV via Internet o per trasmettere del proprio contenuto audio/video. Ciò significa che un utente finale è autorizzato a costruire il suo sistema di trasmissione in diretta. Approssimativamente, con l'applicaszione SopCast un PC può supportare fino a 10.000 peer simultaneamente. *SoP* è l'abbreviazione di *"Streaming over P2P"*. Dal momento che SopCast è un sistema proprietario, non vi è alcun codice sorgente disponibile, pertanto è importante capire le dinamiche di rete e la distribuzione del traffico della rete SopCast, che attualmente mette a disposizione gratuitamente tutti i suoi servizi. I video nel sistema SopCast sono codificati in due tipi diversi di formato: Windows Media Video (WMV) e Real Video (RMVB). Per eseguire SopCast con successo si dovrebbe avere almeno 512Kbps di larghezza di banda e si raccomanda 1Mbps per la connessione internet. La qualità video dei canali dipende da quanti utenti stanno li guardando. L'architettura di molte reti P2PTV può essere considerata come versioni real-time di BitTorrent: se un utente vuole guardare uno specifico canale, il software P2PTV contatta il server tracker per quel canale al fine di ottenere gli indirizzi dei peer che distribuiscono il contenuto video. Sarà poi il tracker a mettersi in contatto con questi peer per scaricare le parti di video, infatti registra l'indirizzo dell'utente in modo che possa essere agganciato ad altri utenti che desiderano visualizzare lo stesso canale. E in questo modo si può notare la sovrapposizione tra i livelli al di sopra dell'Internet regolare.

Figura 4.3: Il client SopCast

Figura 4.4: Il server SopCast

Come abbiamo già detto l'applicazione SopCast ha la funzione di consentire agli utenti di trasmettere il proprio canale: "MyServer" è infatti il nome del canale personale presente nell'interfaccia utente (*Figura j*). Una volta che un utente avvia un canale diventa il provider di origine e comincia a costruire la rete P2P. Prima che un utente possa trasmettere i propri contenuti ha bisogno di registrare un canale dedicato.

Gli utenti possono farlo facendo clic sul link "MySop" sul sito web SopCast. In questo sistema, è possibile registrarsi come nuovo utente, caricare i dati personali, registrare il canale, gestire il canale ecc… . Infine si possono trasmettere i propri contenuti audio e/o video una volta registrato il canale. Come mostra la *Figura k* l'utente deve riempire i campi con i seguenti parametri:

- Channel ID: l'utente può ottenere l'ID del canale dopo essersi registrato. Il nome del canale è facoltativo.
- Server Address: l'indirizzo IP del server in cui il canale registrato.
- Account Name e Password: il nome di account del Broadcaster e la password per accedere al sistema di SopCast.
- Service Port: La porta di servizio di default è 3902. L'utente può cambiarla se lo ritiene necessario.

Ci sono tre modi per un utente SopCast per trasmettere la sua sorgente video: *stream live*, *media directory* e *unico file*. Live Stream alimenta il canale con una sorgente stream live generata dal media server o un media encoder (codificatore multimediale). Se si seleziona media directory, SopCast cercherà tutti i file multimediali supportati nella directory e

sottodirectory e genera automaticamente un file di sop_playlist.spl che utilizzerà per trasmettere tutti questi file multimediali. La selezione a unico file alimenta il canale prelevando un file multimediale da disco locale. SopCast supporta 4 tipi di file: *.asf, *.wmv, *.rm, *.avi, *.mp3, *.spl.

Nell'esperimento che verrà trattato, è stato scelto un singolo file come sorgente video da condividere, ed è anche il metodo più comune di trasmissione per gli utenti. L'emittente è in grado di monitorare la qualità del canale grazie a due parametri: la qualità della fonte (QS-Quality Of Source) e la qualità della rete client (QC-Quality of Client network). Ricordo che il Video on Demand (VoD) di SopCast non è ancora stato implementato.

4.3. Esperimenti e Metodologie

Al fine di approfondire la conoscenza di questo applicativo per la trasmissione di canali televisivi in rete mi sono servito di alcuni esperimenti e misurazioni effettuate da un team di ricercatori provenienti dalla provincia cinese di Qinghai. Attraverso tali dati ho cercato di evidenziare alcuni aspetti importanti e alcune particolarità che a mio parere meritano attenzione.

Come abbiamo già detto SopCast è un software proprietario quindi non rende disponibile al pubblico il codice sorgente che implementa e dunque per comprenderne più a fondo le caratteristiche e il funzionamento occorre trattarlo come una "scatola nera" e analizzare il protocollo SopCast tramite i data files ottenuti da alcune applicazioni di supporto per l'analisi di rete. Una di queste è **Tcpdump** (36), un tool comune per il debug delle reti di computer che funziona da riga

di comando e che consente all'utente di intercettare pacchetti e trasmissioni.

Un'altra applicazione utilizzata è **Wireshark** (37), un software per analisi di protocollo o packet sniffer (letteralmente "annusa-pacchetti") utilizzato per la soluzione di problemi di rete, per l'analisi e lo sviluppo di protocolli o di software di comunicazione. Fornisce una serie di funzionalità molto utili come ad esempio un network analyzer specifico per intercettare pacchetti in rete e presentarli all'operatore in una forma che possa facilitarne l'analisi (human-readable form).

Infine è stato utilizzato **PlanetLab** (38), una piattaforma open-source per lo sviluppo, la distribuzione e l'accesso a servizi su scala planetaria che realizza reti sperimentali ad hoc, i cosiddetti testbed largamente utilizzati dalla comunità di ricerca su tematiche legate al networking e ai sistemi distribuiti.

Negli esperimenti proposti è stato usato PlanetLab per emulare il sistema SopCast in ambiente chiuso potendo così valutare l'intera struttura a sovrapposizione della rete. Si distinguono due tipi di nodi:

- Un source provider: un computer che monta Windows XP, processore Intel Pentium 2.4 GHz e interfaccia di rete 100 FastEthernet che provvede alla generazione della sorgente video. Dunque registra sulla rete SopCast un canale privato che trasmette in loop un filmato della durata di 2 minuti e di dimensione pari a 3,5 MBytes.
- Una serie di nodi PlanetLab: nodi che agiscono come peer SopCast che accedono per visualizzare il canale TV. Ogni nodo utilizza quindi Client SopCast (versione Linux) con controllo a linea di comando e Tcpdump per abilitare il monitoring passivo del traffico.

Sono stati realizzati tre esperimenti che differiscono per la differente dimensione della rete:

1° ESPERIMENTO: La rete prevede la presenza di 50 nodi PlanetLab come SopCast peer. Il primo esperimento è fondamentale per capire la struttura a overlay implementata da Sopcast e per valutare la sua topologia. Su 50 nodi, 45 hanno prodotto un file di log, dunque hanno partecipato attivamente all'intero esperimento.

2° ESPERIMENTO: La rete prevede la presenza di 194 nodi PlanetLab come SopCast peer. Il secondo esperimento riflette le dinamiche di uno scenario più realistico. Su 194 nodi, 153 hanno prodotto un file di log.

3° ESPERIMENTO: La rete prevede la presenza di 441 nodi PlanetLab come SopCast peer. Il terzo esperimento paragonato al secondo è fondamentale per valutare la scalabilità di SopCast. Su 441 nodi, 273 hanno prodotto un file di log.

Occorre precisare alcuni aspetti: la durata di ciascuno dei tre esperimenti è di 40 minuti, ogni nodo PlanetLab è identificato univocamente dal suo indirizzo IP (Internet Protocol) e il motivo percui alcuni nodi sono incorsi in fallimento è dovuto alle limitazioni che PlanetLab impone all'ampiezza di banda e allo spazio sul disco.

4.4. Il protocollo SopCast

Al fine di analizzare il protocollo utilizzato da SopCast ho deciso di effettuare un'analisi mediante l'utilizzo di Wireshark di come avviene la comunicazione tra nodi della rete p2p in base ai pacchetti inviati e ricevuti dai nodi stessi.

Come possiamo notare nella figura sottostante nella prima colonna è indicato l'ID del pacchetto identificato con un numero progressivo crescente, nella seconda colonna la lunghezza in bytes del pacchetto, nella terza e quarta colonna gli indirizzi IP dei nodi sorgente e destinatario del pacchetto, nella quinta colonna compare il protocollo dello strato di trasporto utilizzato e nell'ultima colonna sono indicate le porte dei nodi che comunicano.

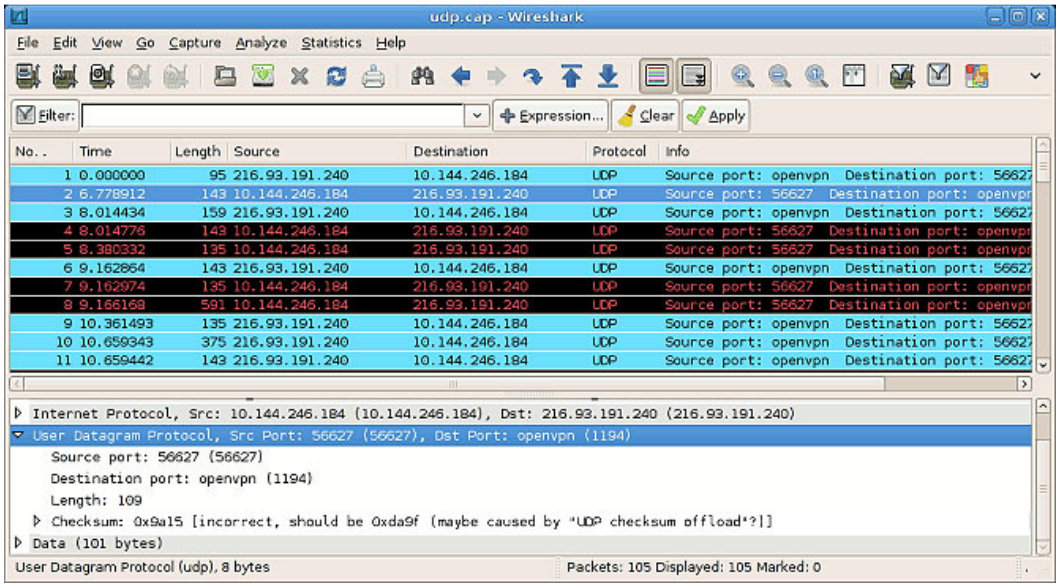


Figura 4.5: Esempio di analisi dei pacchetti inviati e ricevuti dai peer in Wireshark

Già da questa prima analisi ci si accorge che SopCast utilizza il protocollo UDP (User Datagram Protocol) per la comunicazione tra peer.

4.4.1. Identificazione dei pacchetti SopCast

Il traffico in SopCast può essere suddiviso in due tipologie: la consegna di pacchetti di controllo e la consegna del contenuto video:

a. Consegna di pacchetti di controllo

La comunicazione tra due peer in SopCast è sempre inizializzata dall'invio di una coppia di pacchetti di lunghezza rispettivamente 52 e 80 bytes. Una volta che la connessione è stabilita, la coppia di nodi incomincia a scambiarsi vicendevolmente ogni secondo una serie di pacchetti di 42 bytes, tali pacchetti prendono il nome di "*pacchetti keep-alive*".

Inviando questo pacchetto il peer annuncia alla rete la sua esistenza. Il peer che riceve tale messaggio risponde al mittente con un ulteriore pacchetto, questa volta da 28 bytes, mediante il quale segnala l'avvenuta ricezione del precedente keep-alive packet (Acknowledgement). In definitiva se due peer si scambiano questi due tipi di pacchetto significa che sono entrambi consapevoli della propria esistenza e vengono definiti a partire da questo momento in poi "*neighbors peer*", ovvero nodi prossimi o vicini.

Nelle figure sottostanti possiamo visualizzare lo schema di comunicazione fra peer prossimi e possiamo notare anche come sia possibile che un nodo possa perdere il suo prossimo. Infatti nel caso in cui un vicino interrompa la risposta ai pacchetti keep-alive, il peer smette di contattare il suo vicino e si adopera per cercarne un altro.

Communication between two neighbors			
Time (s)	Size (bytes)	Source IP	Destination IP
60.121599	52	152.14.92.59	169.229.50.14
60.197511	80	169.229.50.14	152.14.92.59
60.199382	74	152.14.92.59	169.229.50.14
60.279217	28	169.229.50.14	152.14.92.59
60.738697	42	152.14.92.59	169.229.50.14
60.814576	28	169.229.50.14	152.14.92.59
61.512937	42	152.14.92.59	169.229.50.14
61.588690	28	169.229.50.14	152.14.92.59
		.....	
723.980571	42	152.14.92.59	169.229.50.14
724.057957	28	169.229.50.14	152.14.92.59
724.759458	958	169.229.50.14	152.14.92.59
724.759537	28	152.14.92.59	169.229.50.14

Figura 4.6: Scambio di pacchetti di controllo tra due "neighbors"

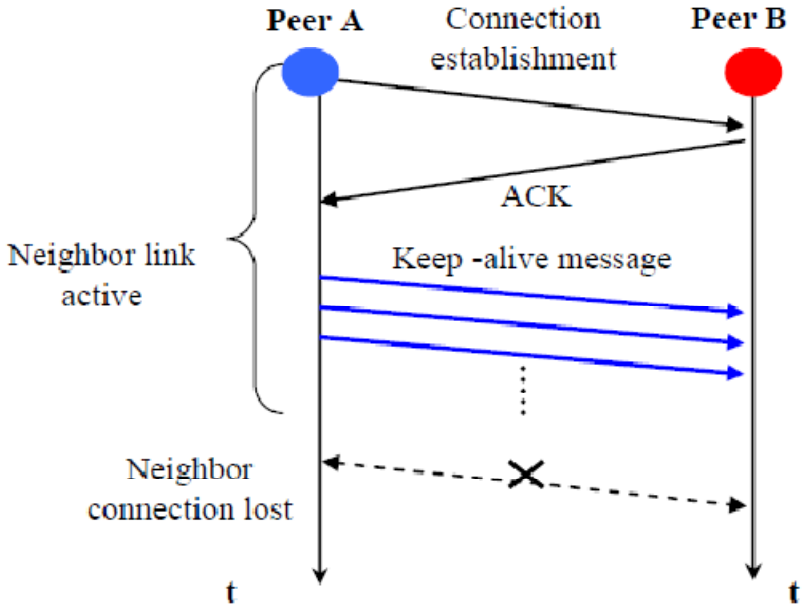


Figura 4.7: Pattern di comunicazione mediante pacchetti di controllo tra "neighbors"

b. Consegna dei pacchetti video

La consegna del contenuto video in SopCast è inizializzata mediante l'invio da parte di un peer di un pacchetto di controllo della dimensioni di 46 bytes con il quale indica la propria richiesta del contenuto video. A questo punto il peer possessore della risorsa incomincia a spedire una sequenza di pacchetti video della dimensione di 1320 Bytes (circa 1 KByte) al neighbor che ne ha fatto richiesta.

Se un peer provvede a fornire contenuto video ad un suo peer vicino si dice essere *genitore* per tale peer ricevente, il quale sarà identificato come *figlio*. All'avvenuta ricezione del pacchetto video da 1320 Bytes il peer risponde al suo sender con un ulteriore pacchetto di Acnowledgement (ACK) di 28 Bytes. Nel caso il figlio necessiti di altre parti di video invierà ulteriori richieste.

Un'approfondita analisi dello scambio di pacchetti fra i peer genitori e figli mostra come la consegna di frammenti video in SopCast sia "*chunk-based*". Infatti la quantità di contenuto TV spedita da un neighbour ad un altro nel contesto della medesima sequenza video in SopCast è suddiviso in blocchi di dimensioni pari a 10 Kbytes. Dunque si sono manifestate numerose sequenze video di dimensioni pari a un multiplo di 10 KBytes (es. 20 KBytes, 30 KBytes, ecc…).

I blocchi richiesti dai peer vengono trattati come un grande datagramma che al fine di scorrere attraverso la rete Internet necessita di essere suddiviso in parti più piccole. SopCast setta un valore massimo per la dimensione di un pacchetto video da trasmettere che è di 1320 Bytes.

Video delivery between two peers			
Time (s)	Size (bytes)	Source IP	Destination IP
61.825380	46	152.14.92.59	169.229.50.14
61.825623	28	169.229.50.14	152.14.92.59
61.826114	1320	169.229.50.14	152.14.92.59
61.826166	28	152.14.92.59	169.229.50.14
61.826244	1320	169.229.50.14	152.14.92.59
61.826289	28	152.14.92.59	169.229.50.14
		.....	
61.830601	1320	169.229.50.14	152.14.92.59
61.830678	28	152.14.92.59	169.229.50.14
61.830744	1320	169.229.50.14	152.14.92.59
61.830812	28	152.14.92.59	169.229.50.14
61.831305	1201	169.229.50.14	152.14.92.59
61.831373	28	152.14.92.59	169.229.50.14

Figura 4.8: Scambio di pacchetti di contenuto video tra due "neighbors"

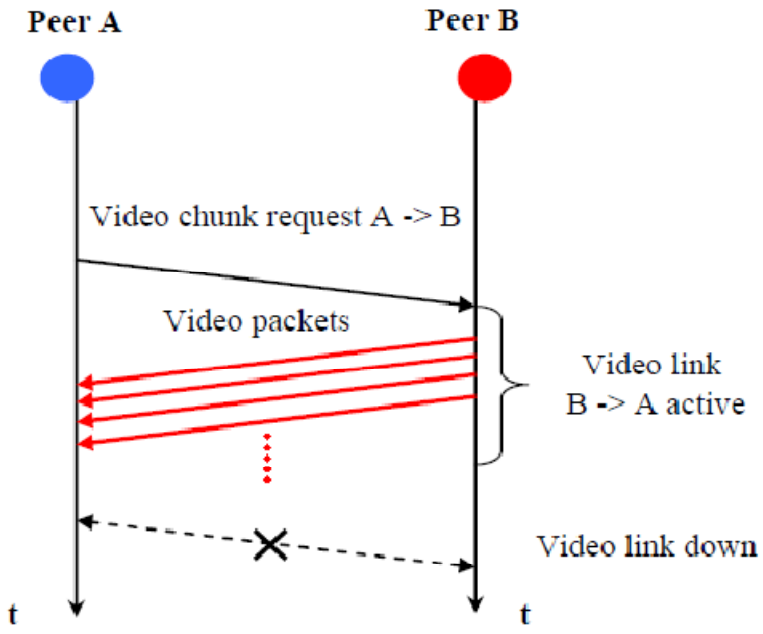


Figura 4.9: Pattern di consegna di contenuto video tra “neighbors”

In base a quanto detto è possibile riassumere in una tabella i pacchetti di più comune utilizzo in SopCast, suddividendoli in tipologie di pacchetti che trasportano contenuti video e pacchetti di controllo.

	Dimensione (in bytes)	Funzionalità
Pacchetti Video	1320	Massima dimensione di un pacchetto video
	377, 497, 617, 1081,1201	Frammenti di video
Pacchetti di Controllo	28	Riconoscimento (Ack)
	42	Messaggio di Keep-Alive
	46	Pacchetto di richiesta video
	52	Pacchetto HELLO per inizializzare le connessioni
	80	Conferma di ricezione del pacchetto HELLO

Figura 4.10: Identificazione dei pacchetti SopCast

4.4.2. Struttura Three-Tier per la spedizione video

Al termine dell'esecuzione del primo esperimento ci si è accorti di un'altra particolarità interessante che riguarda il protocollo SopCast, ovvero il fatto che soltanto 13 nodi su 45 (dei 50 totali previsti) hanno scaricato contenuti video dal Source Provider (SP), il resto dei nodi hanno provveduto contattando a loro volta questi 13 nodi o tra loro stessi.

Le considerazioni dunque sono due: SopCast limita il numero dei figli connessi al SP che richiedono il download di

blocchi video e inoltre SopCast introduce una struttura gerarchica a 3 livelli (Three-Tier Structure) per la consegna del contenuto video.

Come possiamo notare nella figura sottostante, il Source Provider che detiene l'intero il video TV, lo trasmette ad un ristretto numero di peer. I nodi che ricevono il contenuto dal SP vengono definiti "peer di primo livello" (*first tier peers*), mentre tutti gli altri nodi in gioco vengono chiamati "peer di secondo livello" (*second tier peers*). Questi ultimi possono scaricare pacchetti video da peer di primo livello oppure da altri peer di pari livello.

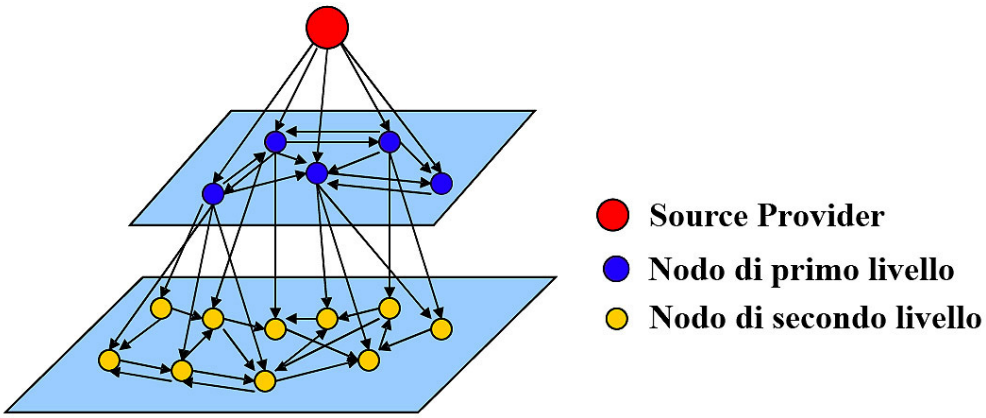
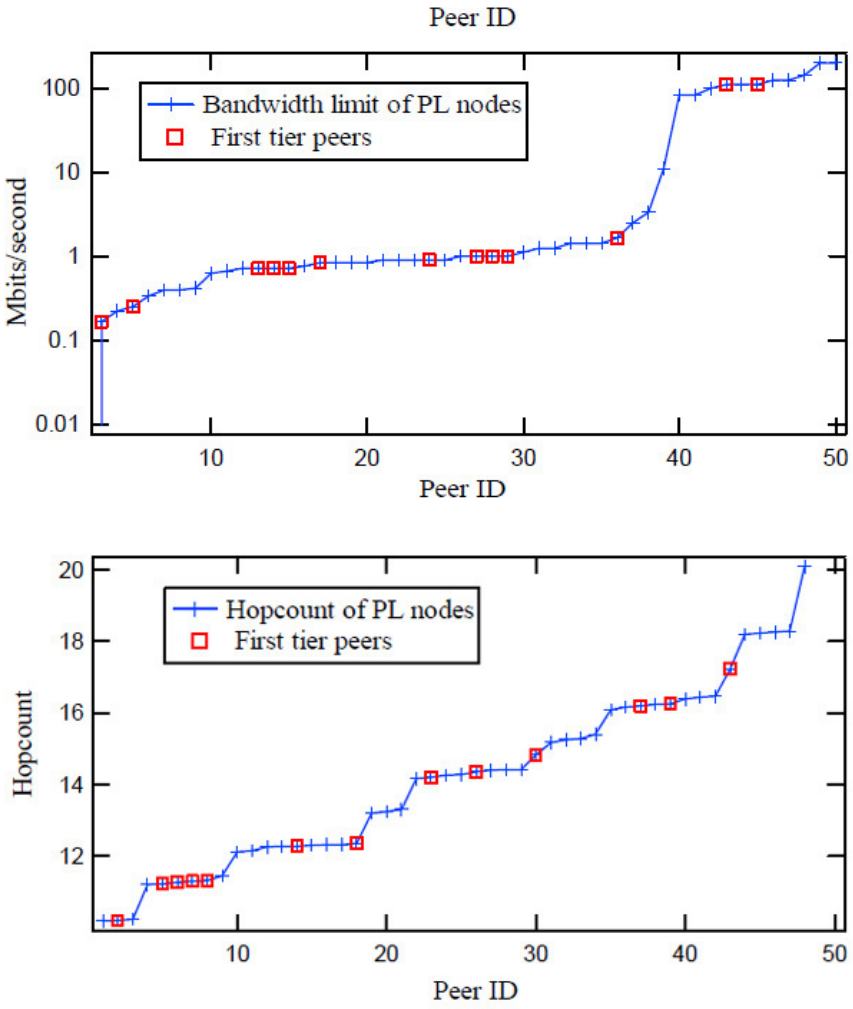


Figura 4.11: Geratchia SopCast a tre strati per la consegna del contenuto video

Ora che è noto come sia strutturata l'architettura di SopCast, è interessante cercare di capire quale strategia SopCast adotti per selezionare i peer di primo livello. Si assegna ad ogni peer un ID unico i tale che $1 \leq i \leq N$, dove N è il numero di nodi PlanetLab previsti dall'esperimento e si vanno a considerare quattro diversi fattori che potrebbero essere critici per un sistema di P2PTV quali:

- *Available Bandwith*, larghezza di banda disponibile;
- *Hopcount*, distanza tra il peer e il SP in termini di livelli;
- *Delay*, ritardo temporale nella comunicazione tra i nodi.
- *Node Locality*, distanza geografica tra i nodi comunicanti;

La figura sottostante mostra come sia possibile graficare questa analisi della performance: sull'asse delle ascisse compare l'ID progressivo crescente dei peer della rete SopCast e sull'asse delle ordinate il parametro di misurazione.



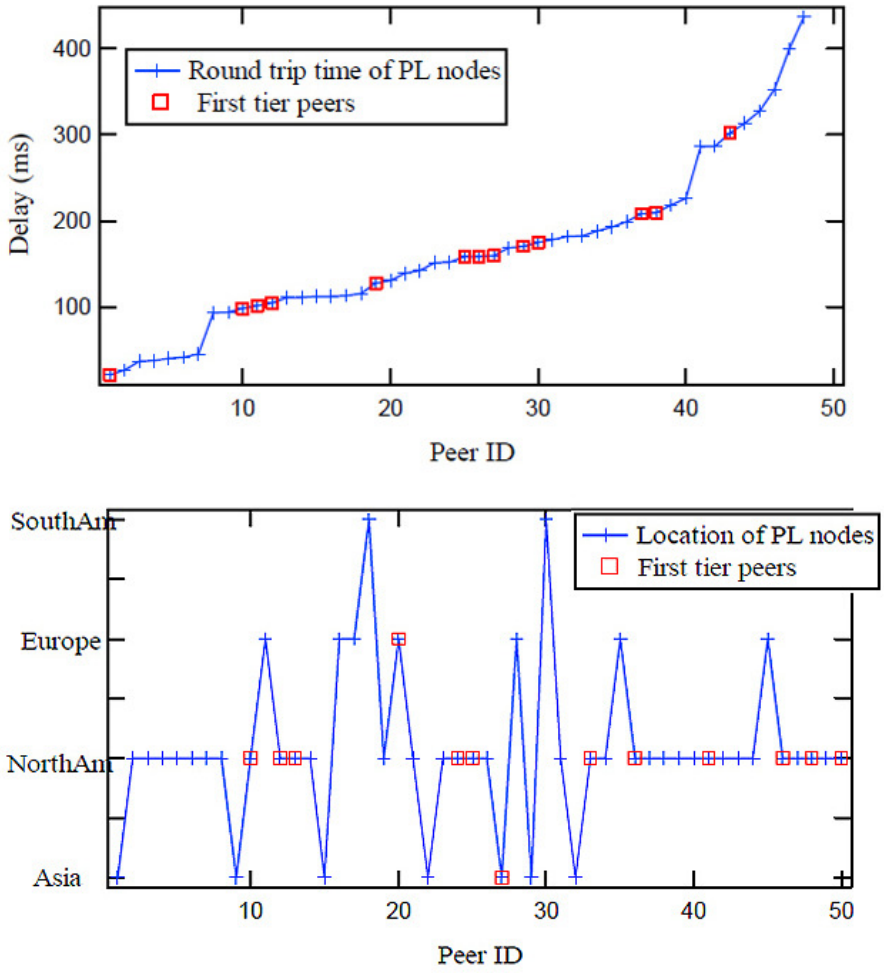


Figura 4.12: Fattori critici per la scelta dei peer di primo livello in SopCast

È possibile notare come il primo grafico mostri l'andamento della curva relativa alla larghezza di banda. Sembra che i peer di primo livello (quelli contraddistinti da un quadrato rosso) non siano necessariamente quelli a possedere maggiore ampiezza di banda. Per quanto riguarda l'hopcount e il delay emerge che i peer tra i peer di primo livello vi sono valori di distanza non omogenei, e stesso andamento anche per la

geografia dei nodi PlanetLab, infatti nonostante ci sia una maggiore concentrazione di nodi provenienti da zone geografiche limitrofe a quelle in cui si è svolto l'esperimento, si può notare la collocazione dei first-tier peer anche in altre regioni del mondo.

Si può concludere che questi quattro fattori critici sono risultati pressoché irrilevanti al fine di comprendere quale possa essere la politica di selezione dei peer di primo livello. A partire da questa considerazione i ricercatori hanno deciso di studiare il processo di accesso di ogni peer.

Dopo che un peer entra nella rete SopCast, il Source Provider incomincia a comunicare con esso inviandogli quello che viene chiamato "*HELLO packet*". Il peer appena connesso replica mediante ACK a tale pacchetto di benvenuto e da ora in poi il SP si predispone all'upload del proprio contenuto video per il peer che ne richiedano la visualizzazione.

Sebbene i peer possano accedere contemporaneamente alla rete, a causa dei non trascurabili ritardi di elaborazione, il SP contatta i peer SopCast in maniera sequenziale nel tempo e prima di incominciare la spedizione dei pacchetti video attende una risposta da questi.

Si deduce quindi che la strategia di selezione dei peer di primo livello sia in linea con il pattern *first-come-first-served* [21] (39). Infatti dal momento che i nodi effettuano il loro ingresso sequenzialmente, i primi a peer andranno a costituire il primo livello, senza alcun tipo di considerazione in merito alla loro specifica ampiezza di banda disponibile. Naturalmente i peer che non rispondo con ack al SP non saranno selezionati. Una

[21] First-come-first-served : Politica in cui le richieste vengono eseguite nell'ordine in cui sono arrivate. Esprime in ambito informatico il concetto di una coda, ovvero la modalità di immagazzinamento di oggetti fisici in cui il primo oggetto introdotto è il primo ad essere utilizzato.

volta che un peer viene selezionato, rimane collegato col Source Provider per tutta la durata dell'esperimento, a meno che non decida spontaneamente di abbandonare la rete.

4.5. Dinamiche topologiche e di traffico in SopCast

La topologia della struttura a livelli (overlay network) di SopCast può essere generata mediante l'utilizzo di peer come nodi e di connessioni come collegamenti fra nodi e in particolare la sovrapposizione dei livelli è costituita da peer decentralizzati, eccetto il Source Provider.

La rete di SopCast è un sistema altamente dinamico in quanto le connessioni tra i nodi coinvolti può interrompersi da un momento all'altro. Cerchiamo in questo paragrafo di indagare la natura dinamica del sistema considerando anche le possibili implicazioni che ciò può portare a livello topologico e siamo interessati anche a scoprire come il carico di rete è distribuito dinamicamente tra i peer attraverso i differenti livelli.

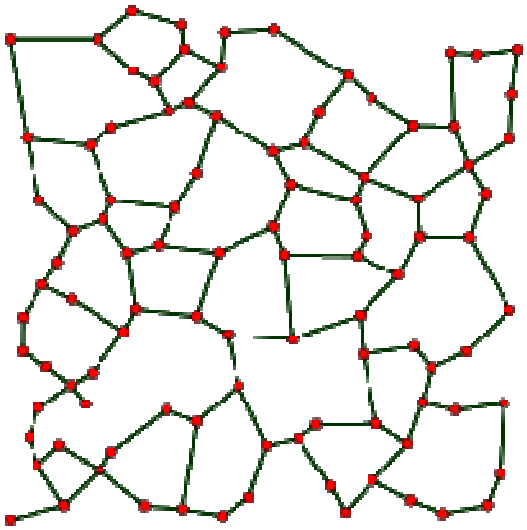


Figura 4.13: Rete di connessioni tra peers dinamici

4.5.1. Architettura Two-Layer

L'architettura di SopCast consiste di due livelli: il "*Neighbor Graph*" G_N e il "*Video Graph*" G_V.
Entrambi i grafi sono formati da connessioni dirette in cui i peer comunicano tra di loro sia in downloading, che in uploading Il Source Provider non è incluso in questa architettura poiché supporta solo un numero ristretto di figli e compie soltanto attività di upload.

Dal momento che non tutti i "nodi prossimi" condividono contenuto video con gli altri peer, possiamo dedurre la seguente relazione lineare tra i due livelli, che indica come G_V sia un sottoinsieme di G_N:

$$\boldsymbol{G_V \subseteq G_N}$$

Neighbor Graph $\boldsymbol{G_N}$: è formato da una serie di nodi che mantengono relazioni con nodi neighbors, i collegamenti tra tali nodi si identificano come "neighbor links". Questi si stabiliscono se tra due pari c'è un regolare scambio di messaggi keep-alive, altrimenti la connessione viene considerata inattiva. Viene indicato inoltre con D_{out} che significa *outgoing degree* (grado d'uscita) il numero di nodi prossimi con il quale un generico peer ha stretto una relazione vi vicinanza nella rete.

Video Graph $\boldsymbol{G_V}$: è formato da una serie di nodi che stanno svolgendo l'attività di trasmissione di blocchi video. Per questi nodi viene valutato il D_{in} che significa *incoming degree* (grado d'ingresso) e indica il numero di genitori che un generico peer può avere nella rete. Mentre il D_{out} indica il numero di figlio che un generico peer supporta nella rete. Un "video link" è

considerato attivo se ci sono pacchetti video in fase di trasferimento da un genitore ad un figlio, altrimenti si indica il collegamento come chiuso.

Possiamo riassumere in figura l'architettura a sovrapposizione tipica di SopCast a due livelli.

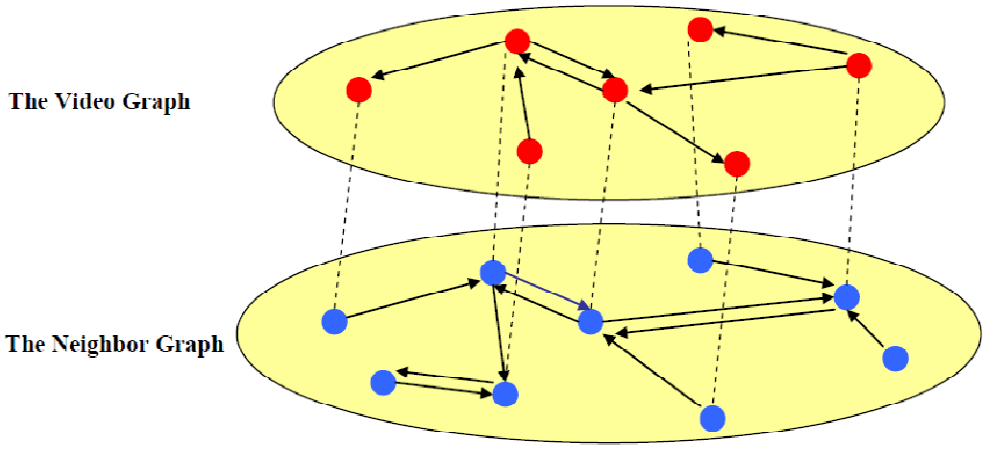


Figura 4.14: Architettura a sovrapposizione a due livelli in SopCast

SopCast P2PTV come ampiamente detto è un software proprietario che non mette a disposizione della community il suo codice sorgente. Ciò rende necessaria un'analisi ad intervalli di tempo di come scorre il traffico tra due peer che comunicano.

Abbiamo parlato di neighbor link interrotto e video link chiuso quando rispettivamente i due peer non comunicano messaggi di keep-alive o di contenuto video. Tuttavia spesso è possibile notare come le connessioni possano cessare temporaneamente. Dopo un certo periodo, il cui range si aggira da qualche secondo a circa un centinaio di secondi, se il link

fosse interrotto i peer neighbor si contattano nuovamente per tentare di riaprire la connessione.

Negli esperimenti svolti in particolare nel secondo e nel terzo, si sono studiate le dinamiche di rete catturando degli snapshots ad ogni intervallo di tempo τ. Definire opportunamente questa variabile temporale è di primaria importanza per il nostro studio, infatti se per esempio fosse preso un intervallo di tempo troppo ampio, i link non più attivi verrebbero comunque catturati portando ad errori nei grafi, e dunque non si rifletterebbero correttamente i cambiamenti dei link di connessione tra i peer.

Andiamo quindi ad analizzare quello che si identifica come *periodo di attivazione* delle connessioni neighbor o video. Per quanto riguarda le prime al fine di determinarne il periodo di attivazione, è stato considerato l'intervallo di tempo che intercorre tra due messaggi consecutivi di keep-alive trasmessi tra due nodi prossimi. Come mostra la figura sottostante in più del 95% dei casi un peer invia due pacchetti consecutivi di keep-alive al suo prossimo in circa 2,5 secondi. Il che significa che se due peer non sono riusciti a contattarsi entro questo intervallo di tempo, la connessione viene considerata terminata. Dunque il periodo di attivazione di un link neighbor è: $\tau_N \sim 2{,}5$ **sec**. Lo stesso approccio si utilizza per i video link, dove però come range di tempo si considera quello che intercorre tra due richieste di contenuto video spedite da un peer ad un suo prossimo. Il periodo di attivazione di un video link è: $\tau_V \sim 2$ **sec**. Se un figlio A richiede un video ad un suo genitore B entro 2 secondi di tempo rispetto alla precedente richiesta, allora il collegamento video tra i due peer è considerato attivo.

L'analisi del periodo di attivazione di neighbor link o video link viene usato per catturare corrette istantanee della rete overlay di SopCast.

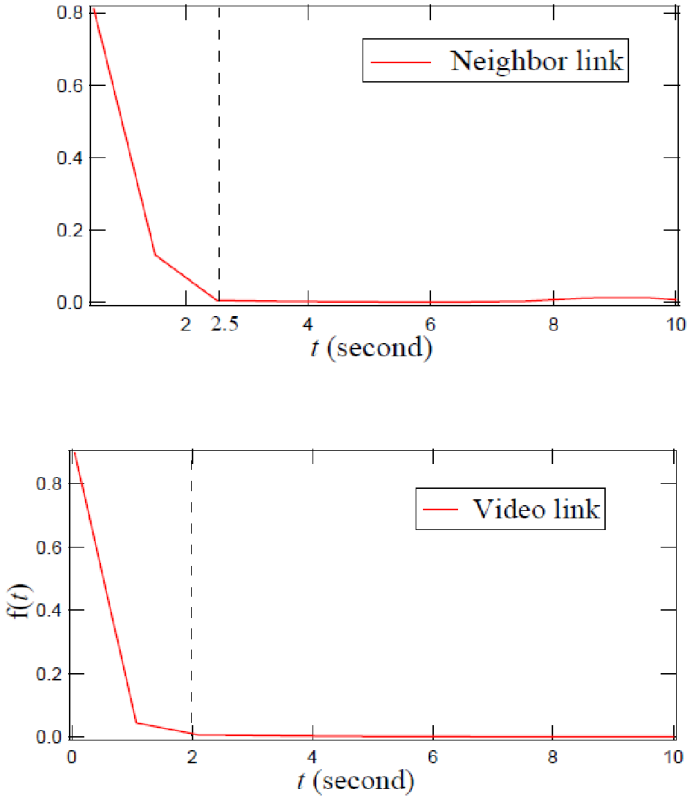


Figura 4.15: Analisi del periodo di attivazione di neighbor e video link

4.5.2. Evoluzione topologica del "neighbor graph"

Le domande che ci si pone ora sono le seguenti: in che tempi riesce un peer SopCast a stabilire una connessione con il maggior numero di peer neighbor della rete? E ancora, qual è la caratteristica del grafo dei peer prossimi?

Si è deciso di catturare delle istantanee del grafo per intervalli di tempo compresi tra [0,T] dove T = 10, 60, 300, 600 secondi rispettivamente. La proposta è quella di esaminare tutte

le connessioni tra peer neighbor accumulate in un preciso lasso di tempo.

Nelle quattro figure sottostanti vengono riportati gli outgoing degree D_{out} di un nodo della rete SopCast, nei diversi periodi di tempo riferiti all'esperimento 1.

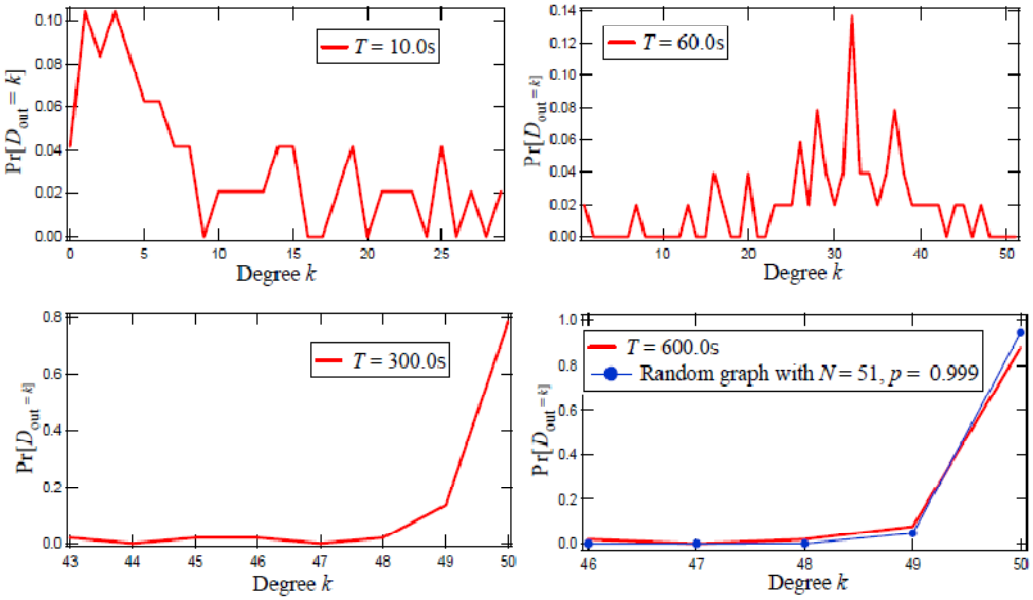


Figura 4.16: Evoluzione topografica del neighbor graph in diversi intervalli di tempo

Si nota che i peer scoprono i loro prossimi abbastanza in fretta nel sistema SopCast. Dopo circa 300 secondi (5 minuti) la topologia del grafo già converge a full mesh, che significa che i peer in SopCast riescono a stabilire connessioni con tutti gli altri peer della rete in tempi relativamente brevi.

Nel prossimo grafico viene illustrata l'evoluzione della media dei gradi uscenti di tutti i peer in funzione del tempo. In

sostanza si delinea la deviazione standard[22] del grado medio d'uscita di un peer neighbor. La media del grado cresce notevolmente nei primi 5 minuti dall'inizio dell'esperimento, infatti come è possibile vedere nella figura v nei casi T=10 sec. e T=60 sec. I peer tendono assumere comportamenti differenti: alcuni cercano di stabilire contatti con molti neighbors, altri invece solo con alcuni di essi. Questo è il motivo percui la deviazione standard nei primi minuti è molto elevata e va affievolendosi col passare dei secondi in seguito all'attività dei neighbors che gradualmente converge fino a diventare pressoché nulla già dopo 5/6 minuti. Ecco infatti che dopo questo lasso di tempo anche la deviazione standard tende a zero.

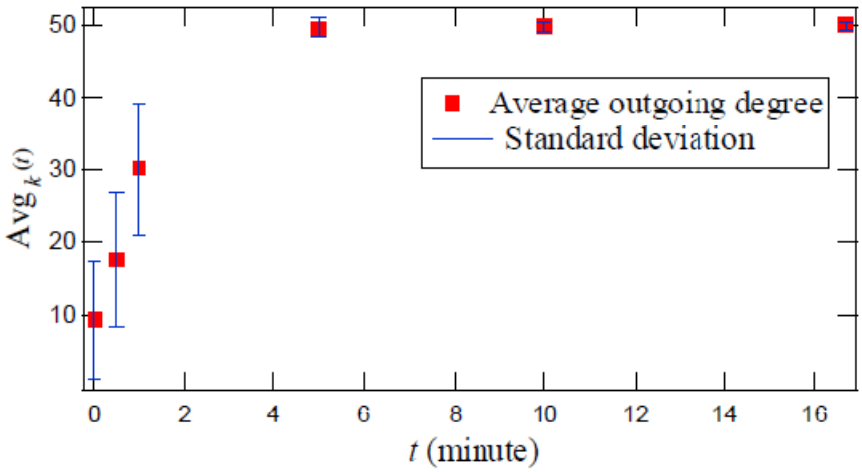


Figura 4.17: Grado medio d'uscita del grafo dei neighbors in funzione del tempo

[22] Deviazione standard:un indice di dispersione delle misure sperimentali, vale a dire è una stima della variabilità di una popolazione di dati o di una variabile casuale. La deviazione standard è uno dei modi per esprimere la dispersione dei dati intorno ad un indice di posizione, quale può essere, ad esempio, il valore atteso o una stima del suddetto valore atteso.

4.5.3. Evoluzione topologica del "video graph"

Per l'analisi dell'evoluzione topologica del grafo video si osservano i peer in un periodo di tempo tale per cui questi possano aver raggiunto la quasi totalità dei peer ad essi prossimi che compongono l'intera rete. Per questa ragione l'analisi la si può far cominciare dopo circa 5 minuti dall'inizio dell'esperimento, poiché si suppone che la rete dopo questo intervallo di tempo risulti stabile. I ricercatori hanno dunque scelto come tempo di misurazione [5 min, 35 min]. Ricordiamo che il periodo di attivazione di un video link è: $\tau_V \sim 2$ sec.

Si è proceduto dividendo il tempo [5 min, 35 min] in periodi di durata 2 secondi, ricavando 900 istantanee di G_V e andando ad osservare le caratteristiche di tutti i D_{out} e D_{in} ottenuti. Il risultato di tale studio emerge evidente nella figura sottostante, i peer SopCast ricevono pacchetti video in media da circa 1,86 neighbor durante i primi 2 secondi di connessione. Solo in rari casi un nodo riesce a stabilire fino a 10 contatti con altri peer al fine di ottenere il contenuto video richiesto.

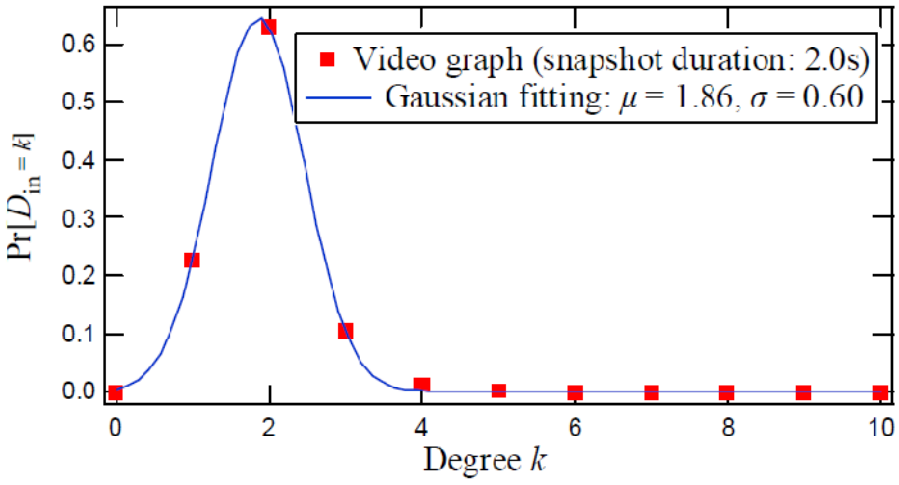


Figura 4.18: Grado di ingresso di un peer dopo 2 sec. dall'avvenuta connessione

Un'ulteriore analisi eseguita dai ricercatori è stata il tentativo di stabilire quanti neighbors un peer contatta nei primi 2,5 secondi. Facendo una media del valore ottenuto in ciascuno dei tre esperimenti, si evince che il numero medio dei neighbors contattati è circa uguale a 38 nodi.

4.5.4. I Super Peers

Un problema critico per i sistemi P2PTV è quello di distribuire equamente il carico della rete tra i peer partecipanti al processo di distribuzione del contenuto. Preferibilmente ogni nodo dovrebbe avere la stessa responsabilità di caricare contenuti per i suoi neighbors, tuttavia è in SopCast non è implementata quello che viene chiamato *principio tit-for-tat* [23].

Infatti un peer nel sistema P2P che stiamo studiando può richiedere una serie di pacchetti video ad un peer suo prossimo senza offrire in cambio alcun blocco video tra i contenuti che possiede e quindi avere intense attività di downloading, ma basse di uploading.

Nella figura sottostante è rappresentato lo througput totale (velocità di downloading e velocità di uploading) di ogni singolo nodo della rete relativa all'esperimento 1.
Precisiamo che nelle ascisse sono stati rappresentati gli ID dei peer ordinati secondo un crescente valore della velocità di upload.

[23] Tit-For-Tat: Un agente utilizzando questa strategia sarà inizialmente un collaboratore, in seguito risponderà con la stessa strategia delle mosse degli avversari: se l'avversario è stato a sua volta cooperativo, l'agente sarà cooperativo, in caso contrario no.

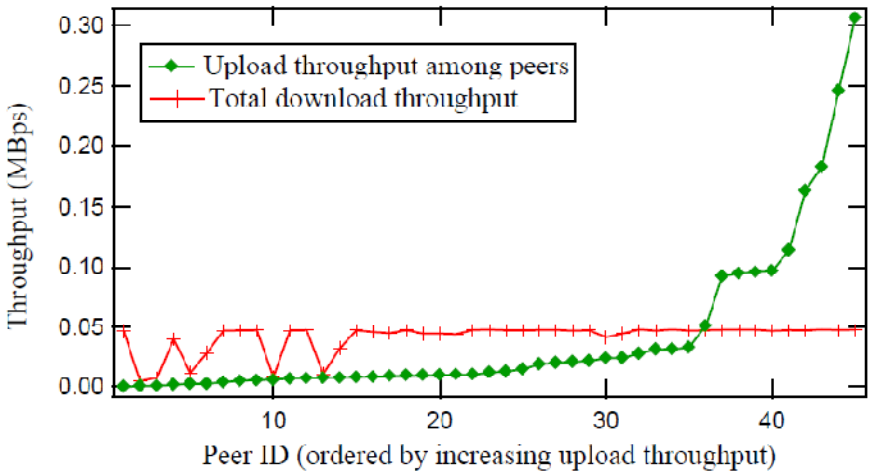


Figura 4.19: Velocità di upload e download di ciascun peer della rete

Si nota come i peers tendano ad uniformare la propria velocità di scaricamento, sebbene vi sia comunque presenza di alcuni nodi che hanno scaricato meno rispetto ad altri, ma ciò è causato dalla minor ampiezza di banda a disposizione di questi peers. Se di uniformità si può parlare per il downloading, non altrettanto si può affermare per l'uploading: infatti soltanto un gruppo ristretto di peers caricano un ingente numero di pacchetti video e li rendono disponibili agli altri neighbors della rete. La maggior parte dei nodi dunque non contribuisce in maniera significativa ad un incremento della propria velocità di upload. Tali peers vengono anche chiamati "*normal peers*" e come detto sono assolutamente la maggioranza tra la totalità dei nodi presenti nella rete. I nodi invece che caricano molto contenuto video (gli ultimi 10 peer che compaiono in tabella, quelli dal 40 al 50) e hanno dunque un'alta velocità di upload vengono anche detti "*super peers*". La presenza di questi super peers apporta un vantaggio alla rete SopCast in quanto allevia l'onere di banda a cui deve provvedere il Source Provider.

Un'ulteriore analisi in merito al traffico dinamico in SopCast è stato effettuata sulla base di una serie di snapshots di durata 2 secondi andando a misurare il coefficiente di correlazione di Pearson[24] tra l'outgoing degree e lo througput di upload.

$$\rho(X,Y) = \frac{E[XY] - \mu_x \mu_y}{\sigma_x \sigma_y}$$

Dove ρ è il coefficiente di correlazione, X e Y sono due variabili casuali. Il numeratore della formula precedente è denominato covarianza ed è un indice del grado con cui le due variabili condividono una comune varianza. Il denominatore $\sigma_x \sigma_y$ invece esprime il prodotto delle due deviazioni standard.

Il significato matematico della formula è insito nel valore che assume il coefficiente: se è uguale a 1 significa che vi è una perfetta correlazione tra le due variabili X e Y, se è uguale a 0 non ve n'è alcuna, mentre se è uguale a -1 significa che c'è una correlazione inversa.

Nel caso della rete SopCast il valore del coefficiente di correlazione di Pearson vale:

$$\rho(outgoig\ degree, upload\ throughput) = 0{,}98$$

Questo valore di ρ mostra con chiarezza una fortissima correlazione tra grado d'uscita di un peer (ovvero numero dei figli che sono connessi al peer in esame) e velocità di caricamento del peer. La conclusione a cui si giunge è che la velocità di upload di un peer e quindi la quantità di blocchi

[24] Coefficiente di correlazione di Pearson tra due variabili aleatorie: è un coefficiente che rappresenta il grado di concordanza o discordanza tra due variabili. Esprime la linearità tra la loro covarianza e il prodotto delle rispettive deviazioni standard.

video che il peer mette a disposizione è tanto maggiore quante più connessioni ha stabilito tale nodo con i suoi figli.

4.5.5. Traffico in funzione della crescita di rete

Restano ancora due questioni di rilievo ai fini di uno studio approfondito delle dinamiche di traffico del sistema SopCast e rispondono alle domande: i peer di primo livello forniscono una quantità di contenuto video in upload simile ai super peers? E ancora, come è distribuito dinamicamente in SopCast il carico di rete tra i peers nei differenti livelli?

Per rispondere a queste domande i ricercatori hanno usufruito delle risposte ottenute analizzando i risultati degli esperimenti 2 e 3.

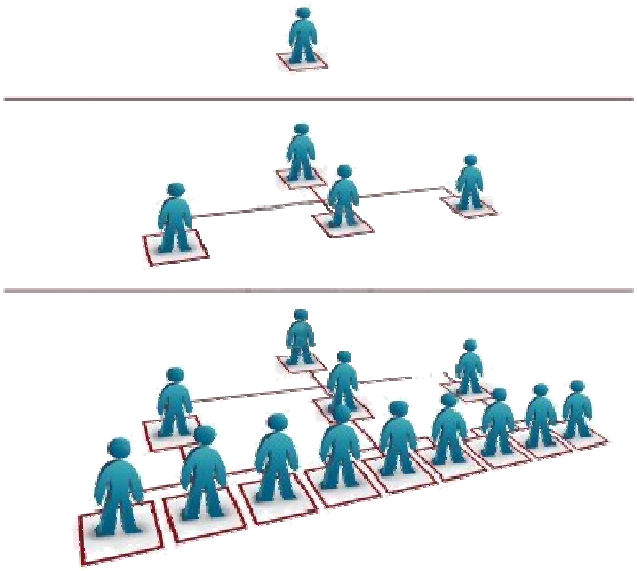


Figura 4.20: Traffico in funzione della crescita di rete

Come già abbiamo detto i peers di primo livello scaricano la maggior parte del contenuto video dal Source Provider, ora proviamo ad capire se e come i peers di secondo livello

preferiscono scaricare da quelli di primo livello o da quelli di secondo livello. A questo scopo come mostrato in figura, si misura il rapporto tra il contenuto video che un peer di secondo livello scarica da un peer di primo livello (curva di colore rosso) o da un peer di secondo livello (curva blu) rispetto all'intero contenuto video scaricato dai peer di secondo livello.

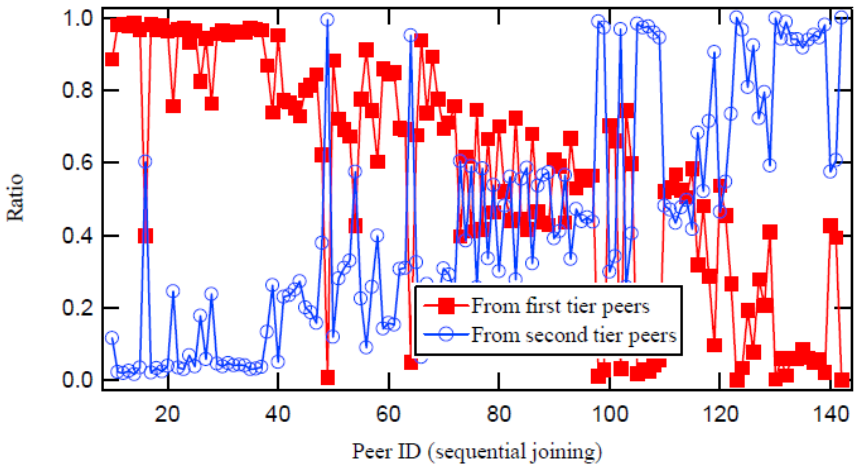


Figura 4.21: Rapporto che valuta le fonti di risorse video dei peer di secondo livello

Possiamo fare alcune osservazioni. Come già asserito, il contenuto video fornito ai peer di primo livello è regolato dal pattern first-come-first-served, ora invece analizzando il grafo in figura, che si riferisce all'esperimento numero 2, possiamo sostenere come i peer di secondo livello con ID minore di 100, ovvero quelli che hanno effettuato l'accesso alla rete per primi rispetto ai successivi, scaricano la maggior parte del contenuto video da peer di primo livello. Mentre quelli che partecipano alla rete da minor tempo (quindi i peer con ID maggiore di 100) sembra che assumano come sorgente video peer di secondo livello, quindi con un rapporto maggiore che man mano tende

ad 1. Per rispondere alla seconda domanda che ci siamo posti all'inizio del paragrafo possiamo fare delle considerazioni circa l'impatto che ha l'ampiezza di banda uscente di un peer rispetto alle sue performance in upload.

La figura sottostante fa emergere chiaramente che i nodi che hanno un'elevata ampiezza di banda a disposizione non sono sempre quelli che rendono disponibile agli altri le maggiori quantità di contenuti video.

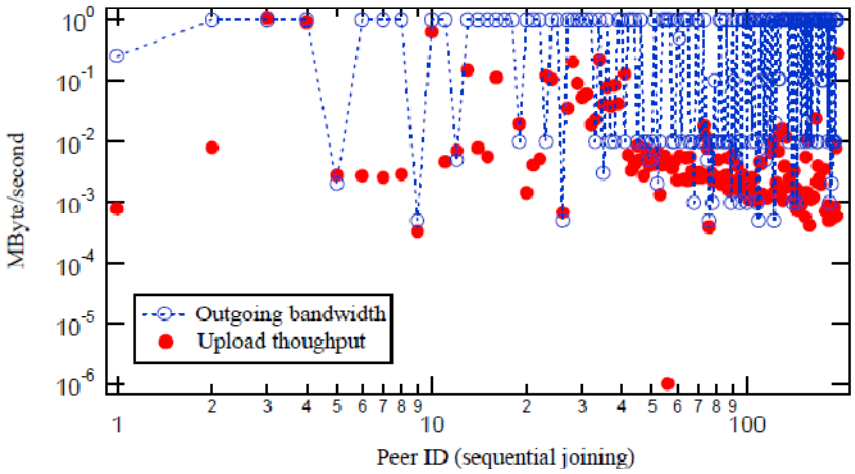


Figura 4.22: Velocità di upload e ampiezza di banda uscente di un nodo

In altre parole, l'ampiezza di banda in download disponibile di un peer non inficia più di tanto sulla sua velocità di upload (40). Il coefficiente di correlazione tra queste due grandezze è stato misurato per tutti e tre gli esperimenti condotti e vale rispettivamente -0.15 , 0.15 e 0.16. Inoltre anche il tempo di accesso di un peer nella rete non influenza le performance di caricamento di un certo peer della rete.

Possiamo quindi affermare che l'algoritmo di scheduling video implementato in SopCast non riesce ad ottimizzare

l'utilizzo delle risorse, ad esempio l'ampiezza di banda e lo scaricamento video dai peer di primo livello, nonché la velocità di upload. Alcuni peer di primo livello infatti risultano di avere un comportamento avido all'interno della rete, nel senso che non provvedono a fornire agli altri componenti una quantità di contenuto grande almeno quanto quella che reperiscono da un'ottima fonte come il Source Provider.

In conclusione SopCast risulta essere non efficiente sia dal punto di vista della distribuzione del traffico di rete che dell'utilizzo delle risorse.

Conclusioni

Il progetto di tesi è incentrato sul mondo live streaming in sistemi peer-to-peer con particolare attenzione all'analisi delle principali caratteristiche di uno dei più famosi software che trasmette contenuti televisivi chiamato SopCast.

Nella prima parte dell'elaborato si è introdotto lo streaming, dalle sue origini ai giorni nostri e sono state evidenziate le principali caratteristiche. Il live streaming è basato sul concetto che un cliente comunica con gli altri per riuscire ad ottenere il file che ricerca. Il file non può essere salvato su hard disk poiché il suo contenuto è disponibile una sola ed unica volta, ovvero nel momento in cui il video/audio viene trasmesso. Il protocollo di riferimento è RTSP che si colloca nel modello ISO/OSI a livello applicativo e quindi in realtà non è propriamente lui ad occuparsi della trasmissione dei dati, sono infatti RTP, TCP e UDP ad occuparsene. Sono state messe in evidenza le differenze che intercorrono tra RTSP e HTTP, benché i due abbiano molti aspetti simili. RTSP si distingue poiché il server utilizzando i protocolli TCP e UDP mantiene lo stato si sessione per permettere la corretta esecuzione della richiesta di un client., inoltre è un protocollo out-of-band, permette la possibilità di inviare richieste client-server e server-client, utilizza sempre l'URI assoluto per identificare le risorse,

permette di mantenere su un singolo host con un unico indirizzo IP diverse strutture di documenti e introduce infine un certo numeri di nuovi metodi, quali describe, announce, setup, get_parameter, set_parameter, redirect, options, pause, play, record e teardown.

Sono state presentate inoltre come modalità di diffusione dei dati, l'unicast, il broadcast, il multicast e il peer-to-peer. Quest'ultimo sistema, che sarà alla base di tutta la tesi, viene utilizzato per la diffusione di contenuto video in tempo reale poiché emerge come esso non richieda l'utilizzo di server di elevate prestazioni e grosse dimensioni. Le architetture peer-to-peer si sviluppano attorno al problema dell'organizzazione dei processi in una rete overlay, ovvero una rete in cui i nodi sono costituiti dai peers e i collegamenti rappresentano i possibili canali di comunicazione. Tali overlay network possono essere suddivisi in due categorie: le reti strutturate, che sfruttano l'architettura ad albero, a multi-alberi e a rete mesh; e le reti non strutturate implementate dall'algoritmo basato sul Gossip. Questi ultimi negli ultimi anni hanno guadagnato il riconoscimento di metodologie adatte alla progettazione di schemi di comunicazione robusti e scalabili.

Si è passati poi a considerare un tema di primaria importanza per lo studio dei sistemi distribuiti come la qualità del servizio e come sia strettamente correlata a quella che è l'ormai celebre filosofia che segue l'intera rete Internet e cioè il best-effort, vale a dire che il sistema garantisce di fare tutto il possibile per portare a termine un'operazione, ma non garantisce affatto che l'operazione verrà compiuta, né in che modo. Abbiamo tuttavia evidenziato due possibili modalità per cercare di fornire comunque una soddisfacente QoS: mediante sovradimensionamento, andando a fornire risorse di rete in

abbondanza abbastanza da soddisfare la domanda di picco attesa, e mediante priorità, amministrando la banda disponibile in modo che i pacchetti a cui deve essere garantita una certa QoS ricevano un trattamento privilegiato. Un altro aspetto da tenere in considerazione è il fatto che un pacchetto che attraversa una rete subisce un ritardo, legato a quattro componenti fondamentali non deterministiche, come il tempo di elaborazione, di trasmissione, di accodamento e di propagazione. Poi abbiamo focalizzato l'attenzione alle P2PTV e abbiamo notato come sia importante in presenza di node churn, ovvero comportamento dinamico degli utenti che accedono alla rete e/o la abbandonano, tentare comunque di garantire un buon servizio. Altri aspetti che influiscono sulla QoS sono il tempo di start necessario al canale, la qualità dello stream video, vista come rapporto tra il numero dei blocchi a disposizione del client e il numero dei blocchi necessari per una perfetta codifica del video; e in ultimo la correzione degli errori causati da possibili perdite di pacchetti durante la trasmissione video. Sono stati introdotti anche possibili metodi di miglioramento della qualità come lo store-and-forward e la trasmissione ridondante dei pacchetti.

Nel terzo capitole della tesi si è discussa una serie di questioni rilevanti nella valutazione della performance delle applicazioni P2P di file sharing. L'analisi è stata effettuata sotto quattro diversi punti di vista: 1)livello utente: è possibile aumentare le prestazioni favorendo la cooperazione tra peers, minimizzando i ritardi di ricezione/trasferimento e velocizzando la ricerca delle risorse. E' importante attraverso opportune politiche di incentivazione cercare di convincere gli utenti a restare nella rete e condividere contenuti, contrastando al massimo un aspetto considerato deleterio per questi sistemi

P2P come il cosiddetto "free riding" secondo il quale un utente egoisticamente abbandona la rete ad avvenuta conclusione delle proprie attività, disinteressandosi completamente della condivisione con gli altri. 2) risorse: da questo punto di vista emerge che un ruolo importante è svolto dalla capacità del servizio che in sostanza è il numero di copie di ciascuna risorsa che la rete può fornire a chi la richiede, dunque maggiore è il numero di repliche, maggiore è la capacità. È inoltre una metrica di tipo dinamico poiché in continua variazione, dal momento che i peers possono accedere e abbandonare la rete ad ogni istante. Abbiamo visto poi come un'analisi in questo senso può essere svolta in fase di regime transitorio, ovvero quella ad inizio sessione e in regime stazionario, quando le richieste delle risorse si stabilizzano e lo througput dei singoli peers si stabilizza. 3) overlay di rete: qui i fattori in gioco sono essenzialmente l'architettura e la topologia della struttura a sovrapposizione del sistema. La valutazione delle prestazioni è dichiarata in termini di numero di messaggi originati dalla query di ricerca per una risorsa, in base al tempo di ricerca, oppure alla probabilità di successo della ricerca stessa. È risultato evidente come sia la scelta della topologia che dell'architettura siano di fondamentale importanza al fine di fornire un servizio che eroghi velocemente le risorse richieste. 4) livello di rete: a questo livello l'interesse si sposta sull'importanza della caratterizzazione del traffico P2P, sulla valutazione dell'impatto che ha sulla rete IP sottostante e la valutazione delle strategie per la gestione del traffico. E' evidente che una mancata corrispondenza tra overlay e topologie di rete fa sì che grandi quantità di traffico P2P vengano inserite in collegamenti di rete più costosi. Siamo passati poi ad analizza quello che è il principio cardine alla base dei sistemi streaming P2P ovvero il

principio di Equità, o Fairness, che ha come obiettivo di supportare lo througput di ciascun nodo della rete. Migliorare le prestazioni in questo contesto significa assegnare in fase di regime transitorio un rate massimo di download, dopodiché in fase stazionaria assegnargliene uno in maniera proporzionale a quanto il nodo ha partecipato alla condivisione. Abbiamo poi parlato di un sottoinsieme dei nodi appartenenti alla rete, detti nodi stabili, che restano sempre connessi e che supportano il Source Provider nella trasmissione dei contenuti informativi. Abbiamo visto come è possibile attraverso un indice valutare il grado di stabilità e abbiamo fornito una possibile soluzione topologica per migliorare le prestazioni. Introducendo infatti un cosiddetto Labeled Tree nel primo livello della struttura ad overlay è possibile riconfigurare opportunamente i gradi di stabilità dei peer in modo tale da ovviare ai problemi di sovraffollamento dei peer stabili. Infine abbiamo trattato il problema della perdita dei dati nella trasmissione e individuato un modo per migliorare le prestazioni del sistema in vista, introducendo un tipo di connessione tra peers, i side link, che permettano il recupero delle informazioni perse attraverso a meccanismi di richiesta e invio dei blocchi mancanti.

L'ultimo capitolo della tesi vede introdotto quello che è considerato una delle principali applicazioni P2PTV, SopCast. È stata fatta una introduzione circa le Peer-To-Peer Television ed è stato sottolineato come la loro struttura sia formata da due strati sovrapposti: uno a livello applicativo detto Overlay, l'altro a livello di rete detto Underlaying Layer. La maggior parte delle architetture utilizzate dai sistemi P2P per il live streaming sfruttano il modello basato sulle reti mesh, proprio come SopCast, il quale è anche secondo alcuni studiosi paragonato dal punto di vista del protocollo che implementa a BitTorrent.

Dopo aver introdotto per sommi capi le principali caratteristiche del protocollo BitTorrent e aver citato le più importanti applicazioni P2PTV ad oggi disponibili, ci si è addentrati nello specifico ad analizzare il software SopCast. Esso ha la funzione di consentire agli utenti della rete di trasmettere il proprio canale rendendo disponibile a tutti il proprio contenuto audio-video, oppure di visualizzare i canale aperti da altri utenti. Al fine di approfondire la conoscenza di questo applicativo si sono utilizzati alcuni esperimenti e misurazioni effettuate da un team di ricercatori cinesi. Mediante tali dati si è cercato di evidenziare alcuni aspetti importanti e alcune particolarità interessanti. SopCast è un software proprietario quindi non rende disponibile al pubblico il codice sorgente che implementa e dunque per comprenderne più a fondo le caratteristiche e il funzionamento occorre trattarlo come una "scatola nera" e analizzare il suo protocollo tramite i data files ottenuti da alcune applicazioni di supporto per l'analisi di rete, quali Tcpdump, Wireshark e PlanetLab. Sono stati realizzati tre esperimenti che differiscono per la differente dimensione della rete e sono stati utilizzati due tipi di nodo per rappresentare la struttura a overlay tipica dei sistemi p2p streaming, come un source provider e una serie di nodi PlanetLab. Mediante gli esperimenti si è potuto identificare chiaramente le funzionalità dei principali pacchetti SopCast. Per quanto riguarda il modello di comunicazione tra peer e la struttura di rete in SopCast, le principali conclusioni che si possono dedurre sono le seguenti: 1) Il traffico può essere suddiviso in pacchetti di controllo e pacchetti video. I pacchetti di controllo offrono diverse funzionalità di coordinamento dell'applicazione SopCast, mentre i pacchetti video permettono il trasferimento dei contenuti TV. 2) La comunicazione tra nodi

in SopCast è decentralizzata. 3) La consegna del video è chunk-based. Ogni video chunk ha una lunghezza pari a 10 KByte. Un peer è libero di chiedere più blocchi da un singolo genitore o da più genitori. 4) Possiamo vedere SopCast come una struttura gerarchica a tre livelli in materia di consegna video. Soltanto un numero limitato di figli può essere collegato direttamente al Source Provider, e questi si definiscono peer di primo livello. Poi ci sono anche peer di secondo livello che scaricano contenuti da peer di pari livello oppure da quelli di di primo livello. 5) SopCast non utilizza un sofisticato algoritmo di selezione dei peer di primo livello. Infatti vengono selezionati sulla base del pattern First-Come-Fist-Served, indipendentemente dalla loro distanza fisica, dalla distanza della rete, distanza dei nodi nel tempo e la loro velocità di banda in upload. Una volta selezionato, un peer di primo livello rimane connesso al Source Provider per la durata dell'intero esperimento, a meno che non abbandoni spontaneamente.

Inoltre si è pensato di modellare la sovrapposizione con un'architettura costituita dai seguenti due livelli: G_N "neighbor graph" e G_V "video graph". L'obiettivo è quello di determinare un intervallo di tempo idoneo per catturare una sequenza di istantanee ai nodi PlanetLab al fine di caratterizzare le proprietà topologiche di questa sovrapposizione dinamica della rete P2PTV. Tale studio ha rivelato che: 1) Il periodo di attivazione di un neighbor link e di un video link è rispettivamente pari a 2,5 sec. e 2 sec. 2) L'analisi del periodo di attivazione di neighbour link o video link viene usato per catturare corrette istantanee della rete overlay SopCast. 3) I peer ricevono pacchetti video in media da circa 1,86 neighbors durante i primi 2 secondi di connessione. 4) Il numero medio di neighbors contattati da un peer entro 2,5 secondi è pari circa a 38. 5)

E'stata osservata l'esistenza di Super Peers, i quali supportano una notevole quantità di peer figli, caricano molto contenuto video e possiedono un'alta velocità di upload.

Sebbene SopCast sia una delle più popolari applicazioni commerciali P2PTV, tuttavia risulta essere non efficiente sia dal punto di vista della distribuzione del traffico di rete che dell'utilizzo delle risorse. Innanzitutto non è ottimizzata l'ampiezza di banda di un peer SopCast in quanto i nodi con una elevata ampiezza di banda, possono non fornire una eguale velocità di upload, quindi la distribuzione del traffico risulta sbilanciata tra i diversi neighbor. In secondo luogo, SopCast impiega un algoritmo molto semplice per la selezione dei peer di primo livello: il pattern first-come-first-served, che prevede che i first-tier peers non abbiano lo stesso ruolo nel caricamento dei contenuti video per l'intera rete. Dunque la performance di upload di questi peers non incrementa in seguito al crescere dei partecipanti alla rete, ignorando la scalabilità del sistema. Un possibile sviluppo e miglioramento del protocollo SopCast dovrebbe prevedere l'esistenza di un algoritmo di selezione dei peer di primo livello più dinamico, che per esempio permettesse la terminazione della connessione di un peer di primo livello qualora, sebbene connesso vantaggiosamente al Source Provider, non contribuisse a fornire agli altri neighbors una buona velocità di upload.

Bibliografia

1. *Mbone Network* http://en.wikipedia.org/wiki/Mbone.

2. **J. Kurose, K. Ross.** *Internet e Reti di Calcolatori.* s.l. : McGraw Hill.

3. *Il Protocollo RTSP* http://unina.stidue.net/Applicazioni%20Telematiche/Materiale/RTSP.pdf..

4. *RTSP Streaming Protocol for P2P Applications* http://unina.stidue.net/Applicazioni%20Telematiche/Materiale/RTSP%20PoliTo.pdf.

5. *Wireshark, The Network Protocol Analyzer* . http://www.wireshark.org/.

6. *Peer2Peer* - http://video.pmi.it/file/vedi/300/peer-to-peer/.

7. **Andrew S. Tanenbaum, Maarten Van Steen.** *Distributed Systems.* s.l. : Pearson - Prentice Hall, 2007.

8. **M. Steiner, D. Carra, E.W. Biersack.** *Evaluating and improving the content access in KAD.* s.l. : Computer Science & Engineering , 2010.

9. *Minimizing node churn in peer-to-peer streaming.* **C. Vassilakis, I. Stavrakakis.** 2010.

10. *On the Performance of multiple-tree-based peer-to-peer live streaming.* **G. Dan, V. Fodor, I. Chatzidrossos.** s.l. : International Conference on Computer Communications, 2007.

11. *Bit torrent - P2P Software Application.* http://www.bittorrent.com/intl/it/features.

12. *PRIME: P2P Receiver-Driven Mesh based Streaming.* **N. Magharei, R. Rejaie.** University of Oregon, 2009.

13. *Peer-to-Peer Membership Management for Gossip-Based Protocols.* **A. J. Ganesh, A. Kermarrec, L. Massoulie.**

14. *On The Quality Of Exoerience of SopCast.* **B. Fallica, yue Lu, Fernando Kuipers, Rob Kooij & Pietr Van Mieghem.** 2008, Vol. The Second International Conference on the next generation mobile applications services and technologies.

15. *Hierarchical Packet Schedulers* http://www.cs.cmu.edu/~hzhang/HFSC/.

16. *QoS-Aware content management in P2P networks.* **M. Meo, F. Milan.** Volendram, The Netherlands : Proceedings of International workshop on hot topics in peer-to-peer systems, 2009.

17. *Performance Modeling of P2P File Sharing Applications* http://www.di.unito.it/~rossano/PUBBLICAZIONI/C30.pdf.

18. *A game theoric approach to provide incentive and service differentiation in P2P network.* **R. Ma, S. Lee, J. Lui, D. Yau.** New York : ACM Sigmetrics, 2004.

19. *A rational model for service rate allocation in Peer-To-Peer.* **M. Meo, F. Milan.** Miami : Global Internet Symposium, 2005.

20. *A performance model for peerm to peer file-sharing services.* **K. Kant, R. Iyer, V. Tewari.** - WWW11, Vol. Poster Session May 2002.

21. *Service capacity of peer to peer networks. In proceeding of INFOCOM 2004.* **Veciana, X.Yang and G. de.**

22. *Modeling Peer-Peer file sharing system.* **Z. Ge, D.R. Figueiredo, S. Jaiswal, J. Kurose, D. Towsley.** San Francisco - USA

23. *The true picture of peer-to-peer-file sharing. http://www.cachelogic.com/research/*. **report, Technical.** - Cachelogic search.

24. *Free-riding, Fairness, and Firewalls in P2P File-Sharing.* **J.J.D. Mol, J.A. Powelse, D.H.J. Epama, H.J. Sips.** - International Conference on Peer-To-Peer Computing, 2008.

25. *Peer-to-Peer multimedia streaming using bittorrent.* **Shah, J-F. Paris & P.** - IEEE International Performance, Computing, and Communications Conference, 2007.

26. **Simpson, We.** *Voice over IP.* s.l. : Focal Press, 2008.

27. *SopCast* - http://www.sopcast.com/.

28. *CoolStreaming* - http://www.coolstreaming.us.

29. *PPStream* - http://www.pps.tv/.

30. *TVAnts* - http://tvants.en.softonic.com/.

31. *PeerCast* - http://www.peercast.com-about.com/.

32. *Zatoo* - http://zattoo.com/restricted.

33. *Tribbler* - http://www.tribler.org/trac.

34. *LiveStation* - http://www.livestation.com/.

35. **Leonardi, M. Mellia.** *Dissecting PPLive, SopCast, TVAnts.* ACM Conext, 2008.

36. *Tcpdump.* http://www.tcpdump.org/.

37. *Wireshark.* http://www.wireshark.org/.

38. *PlanetLab.* http://www.planet-lab.org/.

39. *National Istitute of Standards and Technology.* http://xlinux.nist.gov/dads//HTML/firstcome.html.

40. *Measurement of Commercial Peer-To-Peer Live.* http://www.cs.cmu.edu/~hzhang/papers/PPStreamingMeasurementWRAIPS06.pdf.