GIANNI FENU

Livello rete: principi e architetture

INFORMATICA / 1

ISBN: 88-8467-240-6
Livello rete: principi e architetture

Realizzazione editoriale: CUEC
via Is Mirrionis 1, 09123 Cagliari
Tel/fax 070271573 - 070291201

www.cuec.it
e mail: info@cuec.it

Stampa: Solter - Cagliari
Grafica: Biplano – Cagliari

INDICE

PREFAZIONE.

Sin dagli anni settanta i sistemi informatici hanno cominciato a comunicare tra loro. In quegli anni la semplice attività di input/output verso e da sistemi remoti è stata man mano integrata da livelli di comunicazione più elevati.

L'avvento del personal computer rendendo possibile, in termini economici, la possibilità di elaborazione dei messaggi scambiati ha indotto il contestuale ampliamento del loro tipo, aggiungendo suoni ed immagini ai semplici testi.

La successiva crescita vertiginosa del numero dei personal computer, non più riservati ad attività di impresa ma divenuti strumenti comuni in tutte le case in buona parte del mondo, ha richiesto lo sviluppo di reti di comunicazione sempre più estese e complesse.

Un ulteriore spinta a questo sviluppo è stata determinata dalla diffusione del telefono personale mobile.

Ognuna di queste fasi ha visto la introduzione di nuovi sistemi capaci di trasportare ed indirizzare i messaggi scambiati, gestendo gli intasamenti del traffico ed assicurandone la corretta ricezione.

Tutto questo è possibile attraverso un'architettura sufficientemente definita, che assegna ruoli diversi a componenti diverse e stabilisce regole per le loro interfacce.

Lo sviluppo continuo dei servizi disponibili sulla rete introduce sempre nuovi elementi di complessità e, quindi, la necessità di ricondurla a schemi semplici ed ordinati.

La telematica, che si occupa di queste cose, è una materia vasta ed in continuo divenire, e la sua conoscenza e comprensione non è delle più semplici.

In questo libro, frutto di una ormai lunga attività di insegnamento e di ricerca nel settore, l'autore illustra con chiarezza e notevole capacità di sintesi la vastità dell'argomento, evidenziandone i punti cardine e mettendo a disposizione degli studenti dei corsi di laurea di Informatica uno strumento particolarmente utile.

Partendo dagli algoritmi fondamentali, il testo prosegue trattando il livello rete inquadrato nella più vasta architettura di comunicazione soffermandosi in particolare sulle tematiche della congestione e della Qualità del Servizio.

Francesco Maria Aymerich

CAPITOLO 1.

PRINCIPI.

1.1. INTRODUZIONE.

Il desiderio di capire quali opportunità vengano offerte dalle reti di trasmissione dati e dai relativi servizi è direttamente correlabile al desiderio di capire quegli strumenti di comunicazione dell'informazione ormai inseriti a buon titolo nel nostro quotidiano.

Sono innumerevoli le esigenze che ognuno di noi riesce a soddisfare con l'impiego della "rete".

Nei campi più diversi, da quello finanziario a quello gestionale, da quello commerciale a quello scolastico, non esiste un solo settore della società che non possa dirsi beneficiario del tessuto connettivo digitale messo a disposizione di tutti dalle attuali reti di trasmissione dati.

Tra quelle a estensione geografica l'impiego più ampio è certamente appannaggio di Internet, che nel tempo si è affermata come standard di mercato, de facto, scalzando via via la concorrenza di altri modelli architetturali e protocollari proposti sin dai prima anni '70.

E' possibile oggi affermare che nell'era del mercato aperto e della globalizzazione, il monopolio di settore esiste ed è detenuto saldamente proprio dalla suite TCP/IP.

E' proprio l'impiego di protocolli riconducibili ad una ben specifica pila di riferimento, che, a volte, fa perder di vista il significato da attribuirsi ai differenti

livelli che la compongono, creando commistioni o ambiguità nell'attribuzione dei legami funzione-protocollo-livello.

Il pur tardivo sforzo di standardizzazione prodotto dall'International Standard Organization (ISO) con la creazione del modello di riferimento OSI (Open System Interconnection – 1983), ha avuto il vantaggio di creare una relazione tra funzioni e livelli che, nonostante il consolidarsi del precedente modello Internet, traccia in modo univoco limiti e distinzioni universalmente riconosciute per sistemi aperti di interconnessione (Figura 1).

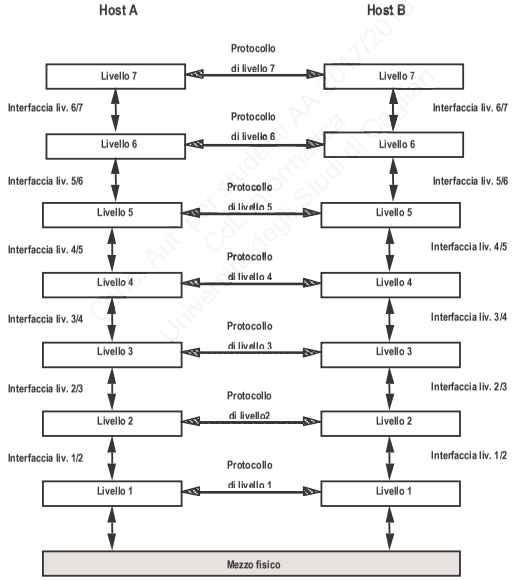

Figura 1. Protocolli, interfacce, livelli, servizi.

1.2. INQUADRAMENTO.

Il modello ISO-OSI si compone di una suite di sette livelli di riferimento (strati, layers) ad ognuno dei quali è possibile attribuire un protocollo di comunicazione, ovvero una convenzione nelle comunicazioni tra livelli pari (omologhi), sapendo che il colloquio, che pur avviene fisicamente tramite il primo di tali livelli (livello fisico) virtualmente consente il collegamento dei soli livelli pari (omologhi). Così il livello di applicazione dell'Host A (elaboratore generico connesso alla rete) colloquierà virtualmente con il livello di applicazione dell'Host B (altro elaboratore connesso alla stessa rete); e ciò avverrà per tutti i livelli a seguire.

Il colloquio fisico sarà tuttavia possibile solo per il livello di connessione reale, livello fisico, mentre per ognuno dei livelli intermedi lo scambio passerà per un trasferimento di informazioni al livello adiacente, attraverso un'interfaccia, e così di seguito per tutta la pila fino a giungere al trasferimento reale, canale di comunicazione del livello fisico (Figura 2).

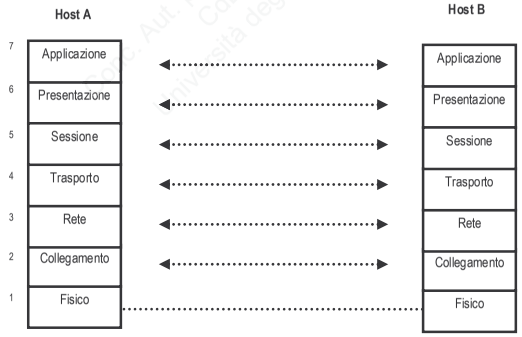

Figura 2. Pila del modello di riferimento ISO-OSI.

E' questo il processo (enveloping) che consente di trasferire informazioni per sistemi aperti, perchè ogni interfaccia si occupa proprio di tener conto dell'ambiente sul quale lavora ogni macchina connessa alla stessa rete, rendendo omologhi i formati scambiati e garantendo l'equivalenza delle informazioni per livelli pari (omologhi).

Proprio nell'ambito delle funzioni attribuite ai livelli del modello ISO-OSI va ricondotto parte dello sforzo classificativo di questo testo.

Ogni livello infatti ha un protocollo, o meglio un insieme di protocolli tipici di quel livello, e delle funzioni specifiche alle quali assolve.

Dovendo descrivere quali siano i compiti di cui si occupa ogni strato della pila, possiamo dire che:

- Il **livello fisico** (physical layer) si occupa del trasferimento del segnale elettrico/ottico su uno o più canali per i quali tale segnale è codificato (bit).
- Il **livello di collegamento** (data link layer) si occupa di delimitare insiemi di impulsi o segnali provenienti dal livello fisico (frame) e di garantirne in qualche modo la correttezza prima di consegnarle ordinatamente al livello successivo.
- Il **livello di rete** (network layer) si fa carico dell'attraversamento dell'intera rete, o meglio sottorete di comunicazione, attraverso la creazione di un percorso (path) che garantisca un inoltro efficiente su nodi noti e su tratte possibilmente non congestionate.
- Il **livello di trasporto** (transport layer) assicura la gestione del trasporto tra i due capi della sottorete (Host A e Host B), garantendone il buon fine in maniera indipendente dall'ambiente e dalla tecnologia sottostante.
- Il **livello di sessione** (session layer) si occupa di gestire l'attribuzione di un token di colloquio tra due HOST, e della sincronizzazione delle comunicazioni tra essi, inclusa la determinazione del punto in cui eventualmente fosse stata interrotta.

- Il **livello di presentazione** (presentation layer) si occupa di definire, a seconda dell'ambiente HOST sul quale opera, un'adeguata conversione dei dati perchè questi siano "leggibili", occupandosi della semantica e della sintassi dell'informazione inviata.
- Il **livello applicativo** (application layer) finalmente consente di visualizzare esattamente il formato dell'informazione originaria, ovvero trasferire un file, ovvero trasmettere un messaggio di posta elettronica, con specifici protocolli.

Ad ogni livello corrispondono dunque specifici **servizi**, insiemi di primitive che assolvono ai compiti di quel livello.

Tali servizi vengono poi erogati al livello superiore (client-layer).

Insiemi di servizi vengono implementati in un **protocollo** specifico del livello. Ogni livello potrà avere più protocolli, diversi tra essi, purché relativi ai servizi definiti nel livello stesso.

Ciò dovrà consentire, comunque, di fornire al livello superiore (client-layer) il servizio atteso.

E' dunque possibile aggregare insiemi di protocolli differenti creando architetture o **suite di protocolli** diverse tra loro ma nel rispetto dei servizi definiti.

Sebbene diverse argomentazioni critiche siano state avanzate sulla eccessiva frammentazione per livelli o in relazione alle modalità di ripartizione delle funzioni, talvolta definita non ottimale, tuttavia il modello ISO-OSI rappresenta il miglior esempio di riferimento nel settore della connessione di sistemi aperti.

Accanto a quella del modello di riferimento ISO altre architetture o suite di protocolli sono state e sono impiegate su reti di calcolatori ad estensione geografica. Tra quelle più significative si ricordano: DECNET (Digital Equipment Corporation NETwork) che per anni ha rappresentato la rete di riferimento nel settore scientifico popolato da sistemi del produttore Digital, SNA (System Network Architecture) rete a forte connotazione gerarchica i cui

standard furono definiti e supportati dal IBM, la suite X.25 (triplaX) rete a pacchetto definita in ambito europeo che rappresenta il modello architetturale più aderente agli standard OSI, e infine la suite Internet o ***TCP/IP*** (Transmission Control Protocol/Internet Protocol) lo standard *de facto* in assoluto maggiormente utilizzato.

Si pensi che agli inizi degli anni '90, l'intera rete italiana della ricerca era equamente divisa tra queste quattro suite protocollari.

La suite TCP/IP ha un suo paradigma di riferimento al modello ISO-OSI, con i livelli del quale non si può dire che la corrispondenza sia biunivoca, ma con il quale, tuttavia, è possibile una comparazione (Figura 3) .

Suite ISO-OSI	Suite TCP/IP
APPLICAZIONE	APPLICAZIONE
PRESENTAZIONE	
SESSIONE	
TRASPORTO	TRASPORTO (TCP-UDP)
RETE	RETE (IP)
COLLEGAMENTO	HOST TO NETWORK
FISICO	

Figura 3. Relazione tra livelli OSI e TCP/IP.

Tra due host afferenti ad una sottorete di comunicazione (Host A, Host B) è possibile dunque instaurare una comunicazione attraverso l'impiego di protocolli comuni implementati anche sui nodi propri della sottorete e che si comportano come commutatori ai fini dell'inoltro (***Router***, Figura 4).

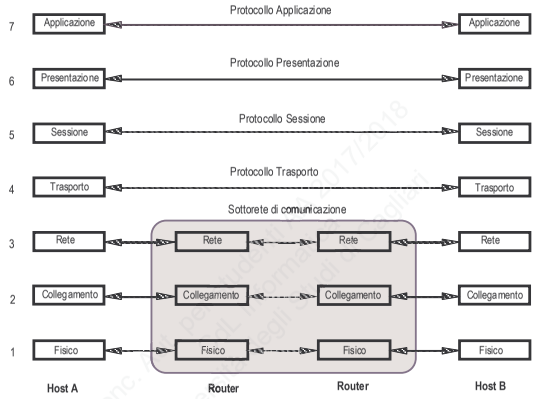

Figura 4. Modello di riferimento ISO-OSI con due nodi commutatori.

La relazione gerarchica all'interno della suite protocollare consente di impiegare nella comunicazione Host A – Host B tutti i protocolli i cui servizi impegnino i primi tre livelli del modello.

Ciò avviene dunque indipendentemente dalla specifica architettura ma nel rispetto delle funzioni assolte da quei servizi e, in definitiva, dai protocolli fino al livello di rete.

E' proprio sulla specificità dei protocolli di questo livello, il ***livello di rete***, che si incentrerà la successiva analisi degli insiemi degli algoritmi di instradamento e

di controllo della congestione fino alle tecniche in grado di garantire la qualità di servizio.

Lungi dal voler rappresentare una raccolta con crismi di esaustività dei protocolli di livello rete, è invece uno specifico obiettivo del testo quello di fornire un panorama agile di classificazione degli algoritmi.

L'indipendenza da uno specifico modello di riferimento e l'esclusione delle tematiche di internetworking e numerazione, sono scelte a garanzia della generalità di trattazione e della snellezza espositiva complessiva.

CAPITOLO 2.

ALGORITMI E PRINCIPI DI ROUTING.

2.1. LIVELLO DI RETE: DEFINIZIONI E SERVIZI.

Il livello di rete si fa carico di trasferire i pacchetti provenienti dal livello di trasporto a partire dalla sorgente fino alla destinazione finale, attraversando tanti sistemi intermedi (***switching points*** o ***router***) della sottorete di comunicazione quanti è necessario secondo le logiche di instradamento adottate.
I principali compiti di questo livello sono:

- conoscere la topologia della sottorete di comunicazione;
- scegliere di volta in volta il cammino migliore (***routing***);
- gestire il flusso dei dati e le congestioni (***flow control*** e ***congestion control***);
- gestire le problematiche derivanti dalla presenza di più reti diverse (***internetworking***).

Nel progetto e nella realizzazione del livello di rete di una architettura di rete si devono prendere decisioni in merito a:

- ***servizi offerti*** al livello superiore (trasporto);

- ***organizzazione interna*** della sottorete di comunicazione.

2.1.1. SERVIZI OFFERTI AL LIVELLO SUPERIORE.

In merito ai servizi offerti al livello superiore, ci sono due tipologie fondamentali di servizi:

- servizi connection-oriented (intendendo servizio connection-oriented un servizio in 3 fasi; instaurazione della connessione, trasmissione delle informazioni, rilascio della connessione);
- servizi connectionless (intendendo come servizio connectionless un servizio ad unica fase; l'invio dell'informazione);

In realtà le decisioni da effettuare riguardano essenzialmente se offrire un servizio affidabile, con uno orientato alla connessione.
Le scelte più comuni sono di offrire ***servizi connection oriented affidabili*** oppure ***servizi connectionless non affidabili***, mentre le altre due combinazioni, anche se tecnicamente possibili, non sono diffuse.

SERVIZIO CONNECTION ORIENTED.

Nei servizi Connection-Oriented per il livello di trasporto, dovranno verificarsi le seguenti condizioni:

- le *peer entities* stabiliscono una connessione, negoziandone i parametri (ad esempio di qualità, di costo) alla quale viene associato un identificatore;
- tale identificatore viene inserito in ogni pacchetto che verrà inviato;
- la comunicazione è bidirezionale e i pacchetti viaggiano, in sequenza, lungo il cammino assegnato alla connessione;

– il controllo di flusso è fornito automaticamente (attraverso alcuni dei parametri negoziati, come ad esempio il dimensionamento di una o più finestre scorrevoli).

La complessità della realizzazione di una rete connection-oriented è ascrivibile ai nodi della sottorete, che si devono occupare del setup delle connessioni e di fornire la necessaria affidabilità.
Attraverso un servizio Connection-Oriented affidabile è possibile ottenere vantaggi quando la Qualità del Servizio è un fattore dominante, ovvero per la gestione di traffico dati in tempo reale (audio/video).

SERVIZIO CONNECTIONLESS.

Le condizioni per la creazione di un servizio connectionless sono:

– la sottorete è giudicata intrinsecamente inaffidabile. Per tale motivo gli host devono provvedere per conto proprio alla correzione degli errori e al controllo di flusso;
– in un servizio senza connessione, i pacchetti sono inoltrati nella sottorete ed instradati in maniera indipendente. In questo contesto, i pacchetti prendono il nome di **datagram**;
– i pacchetti viaggiano indipendentemente, e dunque devono tutti contenere un identificatore (ossia l'indirizzo) della destinazione;
– è inutile inserire equivalenti funzioni di controllo degli errori e del flusso in due diversi livelli;
– la complessità della realizzazione di una rete connectionless risulta negli host, i cui livelli di trasporto devono fornire la necessaria affidabilità e l'orientamento alla connessione.

2.2. PRINCIPI DI COMMUTAZIONE.

La commutazione è la funzione che consente di trasferire l'informazione proveniente dagli host di origine agli host di destinazione variabili di volta in volta secondo le esigenze del servizio.
Tale funzione è espletata da appositi dispositivi collocati ai nodi della rete.
La scelta della tecnica di commutazione da adottare in una sottorete è ristretta a tre opzioni principali, ed influenza la Qualità del Servizio (QoS, Quality of Service).

– commutazione di circuito;
– commutazione di messaggio;
– commutazione di pacchetto.

2.2.1. COMMUTAZIONE DI CIRCUITO.

Il principio di commutazione a circuito è caratterizzato da una comunicazione attraverso tre differenti fasi, ovvero l'instaurazione (*call set-up*), trasferimento dati (*data transfert*), rilascio della connessione (*call clear-down*), tipiche di una connessione connection-oriented.
Nella fase di instaurazione, l'host chiamante (α) invia alla sottorete le informazioni necessarie per l'individuazione del destinatario (β).
E' previsto un ritardo di instaurazione (call set-up delay), variabile con le caratteristiche della rete, dovuto ai tempi di elaborazione per la scelta dei nodi del percorso origine-destinazione (instradamento), ed ai tempi di trasmissione delle informazioni di segnalazione.
Nella figura 5 è possibile individuare la connessione fisica avvenuta dopo la fase di instaurazione.

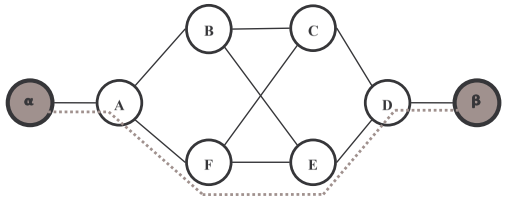

Figura 5. Connessione Fisica in Commutazione di Circuito

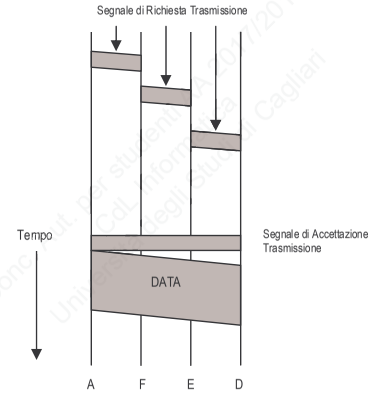

Figura 6. Timing degli eventi nella Commutazione di Circuito.

Durante la fase di trasferimento, la sottorete diventa un'infrastruttura trasparente tra l'origine e la destinazione, ed i canali sono a totale e completa disposizione della comunicazione bidirezionale tra α e β in quanto i dati non subiscono nessuna forma di trattamento intermedio; in effetti, durante la connessione, ed in

maniera esclusiva per le parti in gioco, viene resa disponibile una connessione fisica dedicata.

2.2.2. COMMUTAZIONE DI MESSAGGIO.

Con questa tecnica di commutazione, l'informazione è strutturata in ***messaggi***, ognuno dei quali comprende oltre l'informazione da trasportare, anche altre informazioni aggiuntive quali l'indirizzo del mittente e del destinatario, ed in calce al messaggio opportune forme di verifica della correttezza dei dati trasmessi.

Il trasferimento dell'informazione avviene attraverso la tecnica ***store and forward*** (immagazzinamento e rilascio) per l'impiego dei nodi di trasmissione, e della multiplazione statistica per l'utilizzazione dei rami trasmissivi. Facendo riferimento alla figure 7 e 8, nella commutazione di messaggio, il messaggio inviato dal Host α al nodo A della sottorete viene immagazzinato interamente nello stesso. Attraverso l'esame dell'intestazione, la strategia di instradamento, le regole di priorità, sceglie il nodo successivo (F), per raggiungere il nodo di destinazione (D) e pertanto l'Host β. In caso occupazione del canale di comunicazione da parte di un altro messaggio, questo viene collocato in una fila di attesa. In sostanza il messaggio passa da un nodo a quello successivo, attendendo eventualmente in coda nel caso di ramo occupato.

Ogni nodo intermedio, prima di cancellare il messaggio registrato localmente, attende la conferma dell'avvenuto pervenimento corretto al nodo successivo, e solo ricevuta quest'ultima provvede alla cancellazione locale.

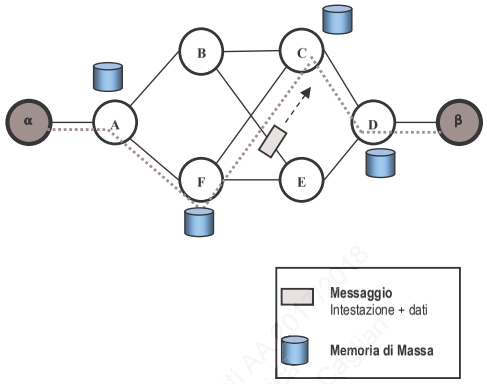

Figura 7. Commutazione di Messaggio.

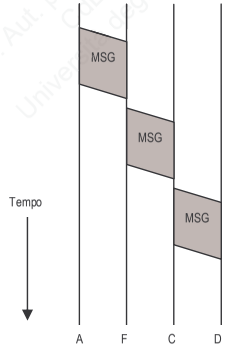

Figura 8. Timing degli eventi nella Commutazione di Messaggio.

Una rete a commutazione di messaggio è caratterizzata da una comunicazione senza connessione (connectionless), in quanto non viene instaurata alcuna relazione, fisica o logica, tra host di origine e host di destinazione.

La sottorete non risulta inoltre trasparente, in quanto i nodi di commutazione operano nei dati le elaborazioni necessarie per effettuare l'instradamento del messaggio.

Il ritardo di trasferimento dei messaggi tra origine e destinazione dipenderà da una serie di fattori, quali la lunghezza dei messaggi, la velocità di trasmissione in ciascun ramo della sottorete, il tempo di attraversamento dei singoli nodi, il numero di nodi e rami necessari per il trasferimento dell'informazione. A causa dei fattori in gioco precedentemente elencati, il servizio di trasferimento delle informazioni in una rete a commutazione di messaggio non può consentire comunicazioni di informazioni interattive.

Inoltre la direzione del messaggio, non prefissata né limitata, ha spesso lunghezze tali che il rischio di corruzione durante l'invio non è infondato.

2.2.3. COMMUTAZIONE DI PACCHETTO.

La naturale evoluzione delle reti a commutazione di messaggio si basa sullo *store and forward* come principio di funzionamento, ma con un diverso utilizzo dei rami trasmissivi attraverso la modalità della multiplazione statistica (ovvero, l'allocazione di più risorse in uno stesso canale in funzione del "carico" di ogni singola comunicazione).

L'informazione viene però segmentata in unità di lunghezza massima predefinita, chiamate ***pacchetti***.

Le reti a commutazione di pacchetti possono funzionare attraverso due modalità differenti:

- datagramma;
- circuito virtuale.

La modalità ***datagramma*** è da considerarsi la diretta evoluzione della tecnica della commutazione di messaggio. L'informazione viene segmentata in una serie di pacchetti chiamati ***datagrammi*** (Figura 9).

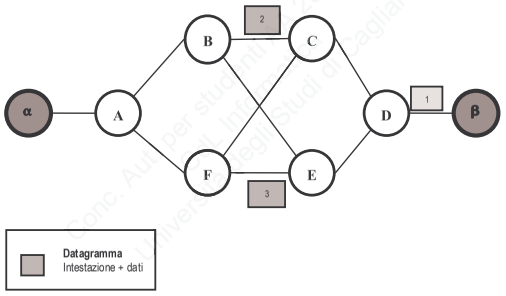

Figura 9. Commutazione di Pacchetto a Datagramma.

Ciascun datagramma è provvisto di intestazione contenente delle informazioni aggiuntive, quali indirizzo di origine e destinazione dell'informazione.
Ogni datagramma si comporta come unità autonoma nel trasferimento attraverso la sottorete; pertanto ciascun pacchetto potrà seguire un percorso differente.

I pacchetti possono essere consegnati fuori sequenza al destinatario, ovvero l'ordine con cui vengono emessi dal mittente non viene conservato nel trasferimento attraverso la rete.
I principali svantaggi della commutazione di pacchetto riguardano da un lato un numero maggiore di informazioni da trasmettere rispetto alla commutazione di messaggio (dovuto alla presenza delle intestazioni per ciascun datagramma), dall'altro la possibilità, frequente, di perdita o duplicazione di pacchetti. Per superare in parte i problemi citati, si ricorre alla tecnica di commutazione di pacchetto a ***circuito virtuale***.
Attraverso tale tecnica, i pacchetti appartenenti alla stessa comunicazione seguono lo stesso cammino (Figura 10). Come si evince dalla figura, tale cammino potrebbe essere diverso per le due direzioni di trasferimento.

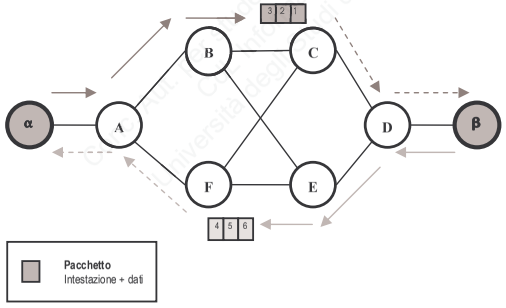

Figura 10. Commutazione di Pacchetto a Circuito Virtuale.

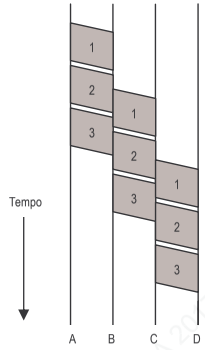

Figura 11. Timing degli eventi nella Commutazione di Pacchetto.

Attraverso la commutazione di pacchetto a circuito virtuale viene garantita la sequenzialità delle informazioni e l'integrità del messaggio a destinazione.
La commutazione di pacchetto a circuito virtuale, al pari della commutazione di circuito, è caratterizzata da una comunicazione attraverso tre differenti fasi, ovvero l'instaurazione, il trasferimento dati, il rilascio della connessione, tipiche di una connessione connection oriented. La differenza tra i due principi di commutazione citati risiede:

- nel fatto che l'instaurazione della connessione avviene tramite l'invio di pacchetti di segnalazione, attraverso i quali avviene la creazione di una serie di canali logici;
- nel trasferimento dati, che avviene lungo un circuito virtuale, ovvero un'associazione logica tra sorgente e destinazione, in luogo di un circuito fisico. I pacchetti appartenenti alla stessa chiamata vengono etichettati con un numero di canale logico, ed instradati in uno dei canali logici

precedentemente creati. Il circuito virtuale viene mantenuto in piedi per tutta la durata della chiamata virtuale;
- nella non trasparenza del flusso dei dati, in quanto ciascun pacchetto deve essere necessariamente elaborato dai nodi della sottorete, al fine di consentire l'instradamento.

2.3. ORGANIZZAZIONE INTERNA DELLA SOTTORETE DI COMUNICAZIONE.

Le sottoreti di comunicazione possono essere raggruppate secondo due insiemi distinti:

a) basato su connessioni ***connection-oriented***. La sottorete stabilisce un circuito, ovvero un cammino fra la sorgente e la destinazione, lungo il quale tutti i router ricordano, in una apposita struttura dati, la parte di loro competenza di tale percorso. Tutti i pacchetti contenti l'ID di tale circuito, verranno instradati, e tutti nello stesso modo, lungo il circuito stesso;
b) basato su connessioni ***connectionless***. I router instradano ogni pacchetto che arriva sulla base del suo indirizzo di destinazione, decidendo di volta in volta come farlo proseguire lungo la sottorete, attraverso l'utilizzo di apposite ***tabelle di instradamento*** (***routing table***). Nel caso di connessioni connectionless ma con servizio connection-oriented, questo livello si occupa di simulare l'esistenza di una connessione al livello superiore, anche se in realtà i pacchetti viaggiano nella sottorete in maniera indipendente, e vengono rimessi in ordine dal livello di rete solo a destinazione, prima della consegna al livello di trasporto.

CONFRONTO TRA SOTTORETI CONNECTION-ORIENTED E CONNECTIONLESS.

	Sottorete Connection Oriented	**Sottorete Connectionless**
Banda Trasmissiva	***Minore*** (piccolo ID in ogni pacchetto)	***Maggiore*** (intero indirizzo di dest. in ogni pacchetto)
Spazio sui router	***Maggiore*** (strutture dati per definire i circuiti fisici o virtuali)	***Minore*** (struttura che definisce i path di inoltro – routing table)
Ritardo per il setup	***Presente***	***Assente***
Ritardo per il routing	***Assente***	***Presente***
Congestione	***Minore*** (risorse allocate in anticipo)	***Maggiore*** (possibile in ogni momento)
Vulnerabilità	***Alta***	***Bassa***

2.4. ALGORITMI DI ROUTING.

La funzione principale del livello di rete è di instradare i pacchetti sulla sottorete, facendo fare loro molti ***hop*** (salti) da un router ad un altro, secondo un percorso (***path***), definito all'interno della rete stessa, tra un host sorgente e uno destinatario.

Un ***algoritmo di routing*** è quella parte del software di livello di rete che determina su quale linea di uscita instradare un pacchetto che è arrivato al router:

- in una sottorete datagram l'algoritmo viene applicato ex novo ad ogni pacchetto, o serie di pacchetti;
- in una sottorete basata su circuiti virtuali l'algoritmo viene applicato solo nella fase di setup del circuito; in tale contesto si usa spesso il termine ***session routing***.

Da un algoritmo di routing ci si attende:

- correttezza (inoltrare il pacchetto nella giusta direzione);
- semplicità (implementazione non troppo complicata);
- robustezza (garanzia di funzionamento anche in caso di cadute di linee di malfunzionamento o blocco del router e di riconfigurazioni della topologia);
- stabilità (convergere, e possibilmente in modo rapido);
- equità (nessun privilegio nell'inoltro);
- ottimalità (scelta della soluzione globalmente migliore).

Purtroppo, gli ultimi due requisiti sono spesso in conflitto fra loro; inoltre, a proposito dell'ottimalità, non sempre è chiaro cosa ottimizzare.
Infatti, ad esempio, si supponga di voler:

- minimizzare il ritardo medio pacchetti;
- massimizzare il throughput totale dei pacchetti.

Si scopre facilmente che questi due obiettivi sono in conflitto fra loro, perché di solito aumentare il throughput allunga le code sui router e quindi aumenta il ritardo: questo è vero per qualunque sistema basato su code gestito in prossimità della sua capacità massima.

Gli algoritmi di routing si dividono in due classi principali:

- algoritmi non adattivi (***static routing***): le decisioni di routing sono prese precedentemente all'avvio della rete, e sono comunicate ai router che poi si attengono sempre a quelle. Questi algoritmi sono adatti per sottoreti di piccole dimensioni, ed hanno il vantaggio di permettere al gestore un controllo totale sui flussi di traffico. Presentano però l'inconveniente della configurazione manuale della sottorete o della sua eventuale riconfigurazione;
- algoritmi adattivi (***dynamic routing***): le decisioni di routing sono frequentemente riformulate, sulla base, ad esempio, del traffico e della topologia della rete. Presentano il vantaggio di un'alta tolleranza alle condizioni di errore, e sono perciò adatti per reti di grandi dimensioni.

Gli algoritmi adattivi differiscono fra loro per:

- come ricevono le informazioni: localmente, dai router adiacenti, da tutti i router;
- quanto spesso rivedono le decisioni: ad intervalli di tempo prefissati, quando il carico cambia, quando la topologia cambia;
- quale metrica di valutazione adottano: distanza, numero di hop, tempo di transito stimato.

METRICA.

Per ***metrica*** si intende il parametro di riferimento per la misurazione dell'attraversamento di una sottorete. Viene associata una variabile ad ogni percorso della sottorete, pertanto, in caso di possibilità di percorsi multipli per la stessa destinazione, viene scelta la route (percorso) con metrica inferiore.

Sono da considerarsi metriche: numero di salti (hops), carico, ritardi, peso delle singole linee, velocità per singola linea, larghezza di banda, costo di comunicazione, affidabilità della route (Figura 12).
Alcuni protocolli possono essere in grado di utilizzare differenti metriche per il calcolo del percorso migliore.

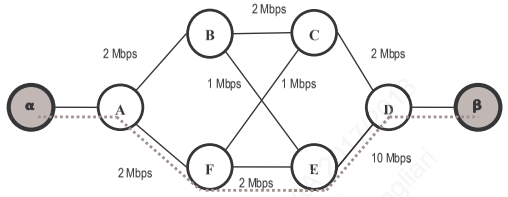

Figura 12. Scelta del percorso basata su metrica larghezza di banda.

E' possibile fare una considerazione generale sul percorso ottimale dei cammini (path), indipendentemente dallo specifico algoritmo adottato per selezionarli. Il ***principio di ottimalità*** dice che se il router j è nel cammino ottimo fra i e k, allora anche il cammino ottimo fra j e k è sulla stessa strada (Figura 13).

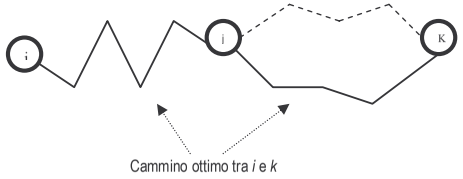

Figura 13. Principio di ottimalità.

Se così non fosse, ci sarebbe un altro cammino (ad es. quello tratteggiato in figura) fra j e k migliore di quello che è parte del cammino ottimo fra i e k, ma allora ci sarebbe anche un cammino fra i e k migliore di quello ottimo. Una diretta conseguenza è che l'insieme dei cammini ottimi da tutti i router a uno specifico router di destinazione costituiscono un ***albero***, detto ***sink tree*** per quel router. In sostanza, gli algoritmi di routing cercano e trovano i sink tree relativi a tutti i possibili router di destinazione, e quindi instradano i pacchetti esclusivamente lungo tali sink tree.

LOAD BALANCING.

La capacità di distribuire il traffico su differenti percorsi, verso la medesima destinazione viene denominata ***bilanciamento del carico*** (***load balancing***). Questo permette una maggiore efficienza sull'utilizzo delle risorse di rete.
Consente inoltre il reinstradamento automatico del flusso dati in caso di anomalie o malfunzionamenti di un percorso. Le due principali modalità di load balancing vengono identificate come:

- *equal-cost load balancing*, che distribuisce il traffico su più percorsi con stessa metrica;
- *unequal-cost load balancing*, che distribuisce il traffico su più percorsi aventi differenti metriche. Il traffico viene distribuito in maniera inversamente proporzionale al costo dei percorsi.

CONVERGENZA.

Viene definito con il termine ***convergenza*** il processo nel quale le tabelle di routing vengono portate ad uno stato di consistenza. Il ***tempo di consistenza*** è il tempo che occorre per l'aggiornamento delle tabelle di routing a seguito di una modifica della topologia di rete.

E' importante notare che si parla di convergenza solo ed esclusivamente con algoritmi dinamici.

2.4.1. ALGORITMI STATICI.

Vengono eseguiti solamente all'avvio della rete e comunque prima del funzionamento a regime della stessa; le decisioni di routing a cui essi pervengono sono poi applicate senza più essere modificate.
Verranno esaminati i seguenti algoritmi statici:

- shortest path routing;
- flooding;
- flow-based routing.

SHORTEST PATH ROUTING.

Un host di gestione della rete mantiene un grafo che rappresenta la sottorete:

- i nodi rappresentano i router;
- gli archi rappresentano le linee punto-punto.

All'avvio della rete, o quando ci sono variazioni permanenti della topologia si applica al grafo un algoritmo per il calcolo del ***cammino minimo*** fra ogni coppia di nodi, che invia tali informazioni a tutti i router.
Per l'individuazione del cammino minimo si può utilizzare quale metrica il numero di hop (salti), la lunghezza dei collegamenti, ovvero il tempo medio di accodamento e trasmissione.

FLOODING.

La tecnica del ***flooding*** consiste nell'inviare ogni pacchetto su tutte le linee del router eccetto quella da cui è arrivato. Presenta l'inconveniente di generare un numero enorme di pacchetti.
Per limitare il traffico generato si può ricorrere a diverse tecniche:

- inserire in ogni pacchetto un ***contatore*** che viene decrementato ad ogni hop. Quando il contatore arriva a zero, il pacchetto viene scartato. Un appropriato valore iniziale può essere il diametro della sottorete;
- inserire la coppia (***source router ID, sequence number***) in ogni pacchetto. Ogni router esamina tali informazioni e ne tiene traccia, e quando le vede per la seconda volta scarta il pacchetto;
- ***selective flooding***: i pacchetti vengono duplicati solo sulle linee che vanno all'incirca nella giusta direzione (per questo si devono mantenere apposite tabelle a bordo).

FLOW-BASED ROUTING.

Questo algoritmo è basato sull'idea di calcolare in anticipo il traffico atteso su ogni linea, da questi calcoli derivare una stima del ritardo medio atteso per ciascuna linea ed infine basare su tali informazioni le decisioni di routing.
Le informazioni necessarie per poter applicare l'algoritmo sono la topologia della rete, la matrice delle quantità di traffico T(i,j) stimate fra ogni coppia (i,j) di router, e le capacità (in bps, ad esempio) delle linee point to point.
Vengono fatte le seguenti assunzioni:

- il traffico è stabile nel tempo e noto in anticipo;

- il ritardo su ciascuna linea aumenta all'aumentare del traffico sulla linea e diminuisce all'aumentare della velocità della linea secondo le leggi della Teoria delle Code.

Dai ritardi calcolati per le singole linee si può calcolare il ritardo medio dell'intera rete, espresso come somma pesata dei ritardi delle singole linee.
Il peso di ogni linea è dato dal traffico su quella linea diviso il traffico totale sulla rete.
Il metodo nel suo complesso, considerata una topologia nota, si basa sui seguenti punti:

- si considera una matrice di traffico;
- si determinano i percorsi che verranno seguiti per il collegamento fra ogni coppia di router;
- si calcola il traffico che incide su ogni linea (che è uguale alla somma di tutti i T(i,j) instradati su quella linea);
- si calcola il ritardo di ogni linea;
- si calcola il ritardo medio della rete;
- si ripete il procedimento con vari algoritmi di routing, scegliendo alla fine quello che minimizza il ritardo medio dell'intera rete.

2.4.2. ALGORITMI DINAMICI.

Gli algoritmi dinamici si adattano automaticamente ai cambiamenti della rete. Non sono eseguiti solo all'avvio della rete, ma rimangono in esecuzione sui router durante il normale funzionamento della rete. Per tale motivo, è utile che il carico complessivo dei pacchetti propri dell'algoritmo sulla sottorete siano un insieme trascurabile del totale del traffico sulla stessa.

DISTANCE VECTOR ROUTING.

Noto anche come algoritmo di Bellman-Ford, la distanza viene espressa attraverso metriche classiche quali distanza e numeri di hop.
Ogni router mantiene una tabella (***vector***) contenente un elemento per ogni altro router. Ogni elemento della tabella contiene:

- la distanza (***distance***, ad esempio il numero di hop, ovvero il ritardo) che lo separa dal router in oggetto;
- la linea in uscita (la direzione, ***vector***) da usare per arrivarci.

Per i suoi vicini immediati il router stima direttamente la distanza dei collegamenti corrispondenti, mandando speciali pacchetti ECHO e misurando quanto tempo ci mette la risposta a tornare. Questa operazione verrà condotta un numero sufficiente di volte, per poter stimare correttamente il tempo di raggiungibilità.
Ad intervalli regolari ogni router manda la sua tabella a tutti i vicini, e riceve quelle dei vicini. Quando un router riceve le nuove informazioni, calcola una nuova tabella scegliendo, fra tutte, la concatenazione migliore

se stesso → vicino immediato → router remoto di destinazione

per ogni destinazione.
Ovviamente, la migliore concatenazione è quella che produce la minore somma di:

- distanza fra il router stesso ed un suo vicino immediato (viene dalla misurazione diretta);

- distanza fra quel vicino immediato ed il router remoto di destinazione (viene dalla tabella ricevuta dal vicino immediato).

Questo algoritmo consente un carico della rete sufficientemente basso, inoltrando i pacchetti echo e scambiando vettori solo con i router adiacenti.
L'algoritmo distance vector routing funziona piuttosto bene, ma è molto lento nel reagire alle cattive notizie, cioè alla caduta di un collegamento. Ciò è legato al fatto che i router non conoscono la topologia della rete.
Infatti, consideriamo questo esempio:

```
A    B    C    D    E  ← Router
*--- --*-- ---*-- ---*--- --*  ← Collegamenti (topologia lineare)
     1    2    3    4  ← Distanze da A
```

Se ora cade la linea fra A e B, dopo uno scambio succede questo:

```
A    B    C    D    E  ← Router
*    *-----*---- -*-- ---*  ← Collegamenti
     3    2    3    4  ← Distanze da A (dopo uno scambio)
```

Ciò perché B, non ricevendo risposta da A, crede di poterci arrivare via C, che ha distanza due da A.

Col proseguire degli scambi, si ha la seguente evoluzione:

```
A    B    C    D    E  ← Router
*    *--- --*-- ---*-- ---*  ← Collegamenti
     3    4    3    4  ← Distanze da A (dopo due scambi)
```

5 4 5 4 ← Distanze da A (dopo tre scambi)

5 6 5 6 ← Distanze da A (dopo quattro scambi)

ecc.

A lungo andare, tutti i router vedono lentamente aumentare sempre più la distanza per arrivare ad A. Questo è il problema del ***count-to-infinity*** (conteggio all'infinito).

Se la distanza rappresenta il numero di hop, si può porre come limite superiore il diametro della rete, ma se la metrica è rappresentata dal ritardo, questo dovrà essere molto alto, altrimenti cammini con un ritardo occasionalmente alto (magari a causa di congestione) verrebbero considerati interrotti.

Sono state proposte molte soluzioni al problema count-to-infinity, ma nessuna veramente efficace.

Nonostante ciò, il distance vector routing era l'algoritmo di routing di ARPANET ed è usato anche in Internet con il nome di ***RIP*** (***Routing Internet Protocol***), e nelle prime versioni di DECnet e IPX.

Nel protocollo RIP, per prevenire il count-to-infinity è stato stabilito un numero massimo di salti pari a 15. Il 16° salto viene considerato infinito.

Con questo limite viene pertanto imposto un limite massimo del diametro della sottorete.

I principali vantaggi di un algoritmo distance vector routing risiedono nella sua facilità di implementazione, e pertanto viene utilizzato in differenti protocolli (tra i quali, oltre al citato RIP, il IGRP, il EIGRP ed il BGP).

I router cooperano per calcolare direttamente le tabelle di instradamento.

Gli svantaggi dell'algoritmo sono però diversi, ed in particolare, oltre al fenomeno dei loop già discusso precedentemente, sono da evidenziare problemi di convergenza (converge alla velocità del percorso più lento e del router più lento), di imprevedibilità di comportamento in reti di grosse dimensioni, ed appesantimento del protocollo in caso di possibili interventi migliorativi allo stesso.

LINK STATE ROUTING.

Il ***link state routing*** supera la lentezza di convergenza del distance vector routing. Ogni router tiene sott'occhio lo stato dei collegamenti fra se ed i suoi vicini immediati (misurando il ritardo di ogni linea), e distribuisce tali informazioni a tutti gli altri.

Sulla base di tali informazioni, ogni router ricostruisce localmente la topologia completa dell'intera rete e calcola il cammino minimo fra se e tutti gli altri.

a. Quando il router si avvia, invia un ***pacchetto HELLO*** su tutte le linee in uscita;
b. in risposta riceve dai vicini i loro indirizzi.

Questi due punti permettono la determinazione dello stato dei collegamenti (***link state***) utilizzando un protocollo di ***Neighbour Greetings*** (saluti ai vicini, ovvero il pacchetto Hello del punto A).

c. Inviando diversi pacchetti ECHO e misurando il tempo di arrivo della risposta e mediando su vari pacchetti si deriva il ritardo della linea. La tabella costruita viene chiamata ***Neighbour Table***;
d. a tale punto viene costruito un pacchetto chiamato ***LSP*** (***Link State Packet***) contenente l'identità del mittente, il numero di sequenza del pacchetto, l'età del pacchetto, la lista dei vicini con i relativi ritardi, la metrica associata, che viene inviato agli altri router;
e. attraverso l'invio e la ricezione di tali pacchetti da parte di ciascun router, viene costruita la topologia dell'intera rete;
f. la fase finale consiste nel calcolo del cammino minimo tra tutti i router.

Ogni router sceglie il percorso migliore per raggiungere la destinazione costruendo l'albero più corto (shortest path tree) avente se stesso come radice ed ognuna delle destinazioni come foglie utilizzando l'algoritmo SPF (Shortest Path First).

La distribuzione dei pacchetti risulta essere la parte più delicata del procedimento, perché errori in questa fase possono provocare una ricostruzione errata dalla topologia di rete da parte dei router.

La distribuzione avviene solitamente utilizzando un algoritmo di flooding, inserendo nei pacchetti le coppie (source router ID, sequence number) per eliminare i duplicati.

Tutti i pacchetti vengono confermati. Inoltre, per evitare che pacchetti vaganti (per qualche errore) girino per sempre, viene decrementata l'età del pacchetto ogni secondo, e quando arriva a zero il pacchetto viene scartato.

Il Link State Routing permette un'alta convergenza, e presuppone inoltre di lavorare su canali punto-punto. In caso di reti broadcast è pertanto necessario modellarle tramite un'equivalente struttura punto-punto.

La più semplice struttura equivalente risulta essere una maglia completa.

La maglia completa presenta alcuni svantaggi, quali il crescere del numero dei collegamenti in maniera quadratica al numero dei nodi di una sottorete, il crescere della complessità dell'algoritmo al numero di collegamenti (E) e nodi (N) con formula E log N, la crescita del numero di database da sincronizzare.

A parte il caso particolare dell'utilizzo dell'algoritmo in reti broadcast, il Link State ha diversi vantaggi, quali la convergenza rapida, maggiore difficoltà alla creazione di loop e comunque facilità di controllo degli stessi, permettendo la gestione di reti di grandi dimensioni.

I router cooperano per mantenere aggiornata la mappa della sottorete.

I principali svantaggi risiedono nella complessità dell'implementazione del protocollo - che ha richiesto alla Digital diversi anni – e la difficoltà di utilizzo in reti broadcast, e quindi in ambito locale.

Il link state routing è molto usato attualmente nella rete Internet attraverso il protocollo ***OSPF*** (***Open Shortest Path First***).

2.5. ROUTING GERARCHICO.

Quando la dimensione della sottorete cresce fino contenere un numero elevato di nodi, mantenere la completa topologia in ogni router risulta essere particolarmente oneroso.

Il routing viene pertanto impostato in maniera gerarchica, in analogia a quanto accade nei sistemi telefonici.

La rete viene divisa in ***zone*** (spesso dette ***regioni***):

- all'interno di una regione i router (detti ***router interni***) conoscono il percorso per il raggiungimento di tutti gli altri router della regione;
- quando un router interno ha necessità di inviare informazioni verso un'altra regione, questo inoltra tali informazioni verso un particolare router della propria regione, chiamato ***router di confine***;
- il router di confine inoltra le informazioni verso il router di confine della regione di destinazione.

Di conseguenza, si possono considerare due livelli di routing (Figura 14):

- un primo livello di routing all'interno di ogni regione;
- un secondo livello di routing fra tutti i router di confine.

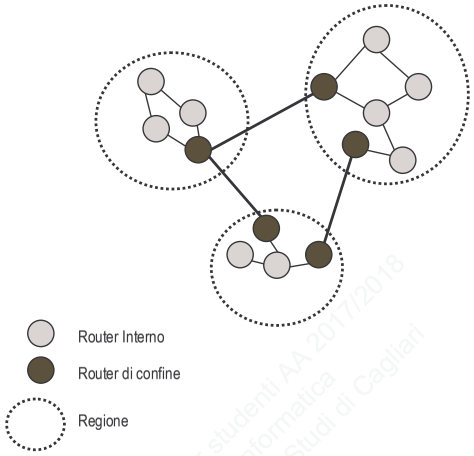

Figura 14. Routing gerarchico.

I router interni mantengono nelle loro tabelle di routing:

- una entrata per ogni altro router interno, con la relativa linea da usare per arrivarci;
- una entrata per ogni altra regione, con l'indicazione del relativo router di confine e della linea da usare per arrivarci.

I router di confine, invece, mantengono:

- una entrata per ogni altra regione, con l'indicazione del prossimo router di confine da contattare e della linea da usare per arrivarci.

Non è detto che due livelli siano sufficienti. In tal caso il discorso si ripete su più livelli.

2.6. BROADCAST ROUTING.

Alcune applicazioni necessitano dell'invio delle informazioni a molti o a tutti gli host presenti in una sottorete. La trasmissione contemporanea di un pacchetto a tutte le destinazioni viene chiamata ***trasmissione broadcast***. Le metodologie per l'ottenimento di questo risultato sono svariate.

La metodologia più semplice consiste nell'invio di un pacchetto distinto a ciascun destinatario. Possiede il vantaggio di un basso utilizzo di banda, ma obbliga il sorgente a possedere una lista completa dei destinatari.

Una seconda, semplice, metodologia consiste nell'utilizzare il meccanismo flooding, già visto per reti punto-punto. Gli svantaggi sono pertanto gli stessi, ovvero generazione elevata di pacchetti e largo consumo di banda.

Un terzo meccanismo consiste nell'utilizzo del ***multidestination routing***. Ogni pacchetto è dotato di una lista di destinazioni. Il router che riceve il pacchetto controlla tutte le destinazioni e determina l'insieme delle linee di trasmissione richieste per inoltrarlo.

A questo punto il router genera una copia del pacchetto che inoltra in ogni linea creata, includendoci esclusivamente le destinazioni su tale linea. Attraverso tale metodologia, dopo un numero sufficiente di salti, ogni pacchetto conterrà esclusivamente un unico indirizzo di destinazione, e sarà trattato pertanto come pacchetto normale.

Un quarto algoritmo utilizza il sink tree del router di trasmissione, ed utilizza lo spanning tree per inoltrare i pacchetti. Lo ***spanning tree*** consiste nel sottoinsieme di percorsi comprendenti tutti i router ma senza cicli (Figura 15). Ogni router appartenente allo spanning tree può copiare un pacchetto broadcast in arrivo su tutte le linee, ad eccezione di quella di ingresso.

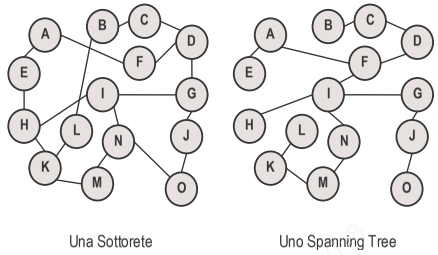

Figura 15. Spanning Tree.

Questo metodo ha il vantaggio di un eccellente utilizzo della banda disponibile, e della minima generazione di pacchetti necessari per il compimento del lavoro. Lo svantaggio principale è la necessità della conoscenza da parte del router di uno spanning tree, cosa impossibile per alcuni algoritmi, quali ad esempio il distance vector. Per ovviare a questo problema, si può utilizzare una metodologia chiamata ***reverse path forwarding***.

Il router, al ricevimento di un pacchetto broadcast, verifica se questo è giunto dalla linea utilizzata per inviare i pacchetti alla sorgente della trasmissione broadcast. In caso affermativo, esiste una alta probabilità che il pacchetto broadcast abbia seguito il percorso migliore per giungere al router, e che perciò sia la prima copia arrivata. Pertanto il router invia il pacchetto verso tutte le linee, ad eccezione di quella entrante.

Viceversa, nel caso in cui il pacchetto broadcast non sia giunto dalla linea utilizzata per inviare pacchetti alla sorgente, quest'ultimo viene scartato in quanto supposto duplicato.

I vantaggi del reverse path forwarding risiedono principalmente nella sua efficienza e nella sua facilità di implementazione; come visto, non richiede

inoltre, da parte dei router, la conoscenza di uno spanning tree o di alcuna mappa.

2.7. MULTICAST ROUTING.

La possibilità di inviare pacchetti a sottoinsiemi (gruppi) di destinatari di un insieme viene chiamato ***multicast***, e l'algoritmo utilizzato per l'instradamento di tali pacchetti viene chiamato ***multicast routing*** (Figura 16).

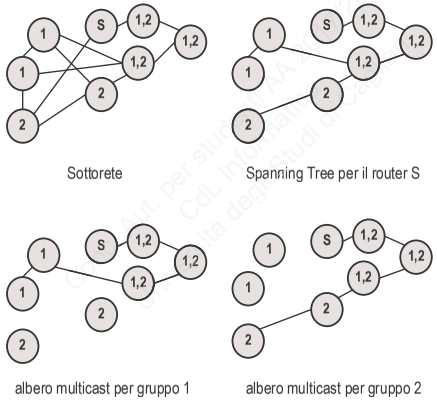

Figura 16. Multicast Routing.

La trasmissione multicast prevede la gestione dei gruppi, ovvero la possibilità di creazione e distruzione dei gruppi, e la metodologia in ingresso ed uscita dai gruppi.

Per ottenere una trasmissione multicast, ogni router elabora una spanning tree che copre tutti gli altri router.

Quando un processo invia un processo multicast ad un gruppo, ogni router rimuove tutte le linee dello spanning tree che non conducono agli host del gruppo, inoltrando pertanto i pacchetti esclusivamente attraverso lo spanning tree appropriato.

2.8. INTERNETWORKING.

L'esistenza di differenti architetture di rete genera notevoli problemi non banali, tra i quali, è opportuno ricordare:

- la difformità nei servizi offerti;
- la difformità nei formati dei pacchetti;
- la difformità nei formati degli indirizzi;
- la difformità nelle sottoreti dei meccanismi di controllo dell'errore;
- la difformità nelle sottoreti dei meccanismi di controllo della congestione;
- la difformità nella dimensione massima dei pacchetti.

La comunicazione tra reti con differenti caratteristiche (Figura 17) è realizzata attraverso dispositivi chiamati ***router multiprotocollo***.

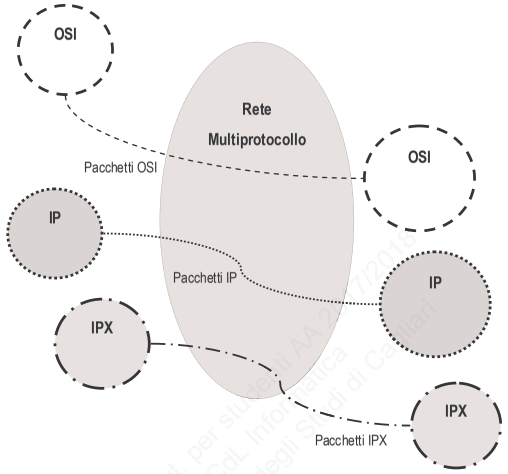

Figura 17. Internetwork basata su router multiprotocollo.

Una seconda tecnica in grado di risolvere il problema viene chiamata **Tunneling**, utilizzata nel caso – frequente - che host sorgente e host di destinazione utilizzino lo stesso tipo di rete, ma separati da una rete con caratteristiche differenti (Figura 18).

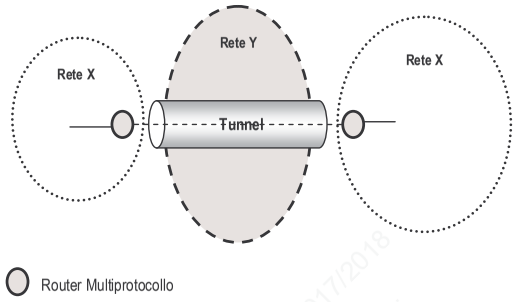

Figura 18. Tunneling.

In questo caso la rete di tipo Y non è dotata di router multiprotocollo. Invece, un router designato in ciascuna delle due reti di tipo X è multiprotocollo, e incapsula i pacchetti delle reti di tipo X dentro pacchetti di tipo Y, consegnandoli poi alla rete di tipo Y. In questi pacchetti ci sono due buste di livello di rete.

2.9. STRUTTURA DI UN GENERICO PACCHETTO.

Pur senza entrare nella specificità della ripartizione dei campi previsti da suite protocollari differenti, è utile fare un breve cenno a quali campi debbano essere comunque contenuti nel pacchetto inoltrato lungo una sottorete di comunicazione:

- *campi sorgente* e *destinazione:* soggiacciono a regole di indirizzamento specifiche; saranno presenti tra i primi byte del pacchetto, e possono

rappresentare sia un indirizzo WAN univoco che un indirizzo di sottorete locale;
- *campo lunghezza*: un campo che specifica la dimensione del pacchetto nel suo insieme o della sola intestazione dello stesso;
- *campo protocollo:* specifica quale è la suite di protocolli tipici della sottorete attraversata;
- *campo percorso:* individua particolarità relative a circuiti predefiniti o permanenti di attraversamento della rete*;*
- *campo versione-servizi*: individua particolarità di impiego di protocolli specifici, sperimentali, o relativi a etichettature di instradamento, ovvero servizi particolari erogabili;
- *campo frammentazione:* riporta informazioni sul progressivo di frammentazione del pacchetto, indicando anche l'eventuale non frammentabilità, o la non ulteriore frammentabilità;
- *campo Time To Live:* definisce il tempo di vita del pacchetto, allo scopo di evitare che un pacchetto non correttamente inoltrato continui a circolare all'infinito nella sottorete; tale campo di norma rappresenta il numero massimo di secondi di vita del pacchetto dal momento della sua generazione; il valore viene decrementato dai router che trova lungo il suo percorso; il pacchetto viene eliminato, quando il valore raggiunge lo zero;
- *campo checksum:* rappresenta l'algoritmo di controllo dell'intestazione o dell'intero pacchetto e che viene calcolato di norma alla sorgente ed alla destinazione;
- *campo dati*: dipende dalla dimensione più o meno flessibile di byte accettati dalla sottorete.

Può essere presente inoltre un campo *timestamp*, che fissa data e ora di creazione o di transito del pacchetto su un nodo della sottorete.

CAPITOLO 3.

CONTROLLO DELLA CONGESTIONE.

3.1. CONGESTIONE.

Uno dei problemi che si incontrano frequentemente nella gestione delle sottoreti di comunicazione, relativamente all'occupazione dei canali trasmissivi, viene definito come **congestione**, ovvero una situazione in cui la rete, o una sua parte, perde in prestazioni a causa della presenza di un numero di pacchetti superiore alla quantità gestibile su quella rete o su quella parte di rete.

La soluzione a questo problema viene chiamata **controllo della congestione**.

Il controllo della congestione si occupa di definire criteri solutivi tali che la sottorete sia in grado di trasportare il traffico richiesto, esaminando il comportamento dei router, il processo di memorizzazione e ritrasmissione all'interno dei router, degli host afferenti alla sottorete e di tutti i fattori che tendono a diminuire la capacità della sottorete di inoltrare i pacchetti.

3.1.1. CONTROLLO DELLA CONGESTIONE.

La congestione è una condizione di ritardo critico causata dalla presenza di un numero eccessivo di pacchetti in uno o più router, in una o più interfacce di un router.

Ciascun router possiede un buffer di una certa grandezza, in grado di conservare i pacchetti giunti ma non ancora spediti o reinoltrati.

Più correttamente bisognerebbe dire che di buffer ne esiste uno per ogni interfaccia di linea.

Durante il fenomeno della congestione, la quantità di pacchetti in ingresso nel router è superiore alla quantità di pacchetti in grado di essere instradati dallo stesso, aumentando la lunghezza della coda, e, di conseguenza, il ritardo di trasmissione.

A causa del meccanismo di *Timeout and Retransmission,* tipico della maggioranza dei protocolli di routing, al timeout generato dal ritardo di trasmissione del router, il nodo risponde ritrasmettendo i pacchetti non inoltrati, aggravando quindi il livello di congestione globale della rete o del segmento di rete. La condizione critica della congestione derivata dall'aumento esponenziale del traffico viene chiamata **Congestion Collapse**, ovvero collasso dovuto alla congestione.

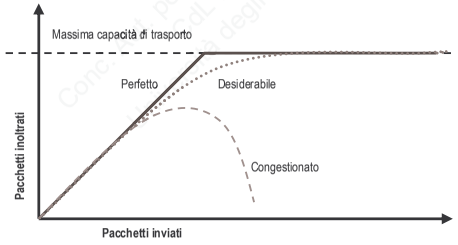

Figura 19. Effetti della congestione.

La congestione in un router può derivare da diversi fattori, ad esempio dimensione dei singoli buffer, numero limitato di buffer nel router, processore troppo lento nel router, linea di trasmissione troppo lenta.

3.2. POLITICHE DI CONTROLLO DELLA CONGESTIONE.

Ogni coda interna di un router viene gestita da una politica di *congestion avoidance* che caratterizza il tipo di reazione che la stessa ha al verificarsi di una congestione. Nel corso di questi anni sono state elaborate diverse politiche, mano a mano più sofisticate, per fronteggiare il sempre più serio problema della congestione.

Gli algoritmi di controllo della congestione possono operare essenzialmente all'interno degli host della rete di comunicazione, gestendone le comunicazioni (*source algorithms,* algoritmi di sorgente), ovvero a livello *link* (collegamento), attraverso l'individuazione della congestione da parte dei router, e la successiva segnalazione del problema alla sorgente (feedback).

L'approccio al problema della congestione si può ricondurre alle due categorie:

- l'***open loop*** (basato su una politica di prevenzione delle congestioni);
- il ***closed loop*** (basato su una politica di controllo della congestione attraverso azioni opportune).

Esistono numerose tecniche di controllo della congestione. Pertanto in questa sede verranno trattate le principali.

3.2.1. CHOKE PACKET.

In questo approccio, di tipo closed loop, è previsto che un router tenga d'occhio il grado di utilizzo delle sue linee di uscita. Il router invia all'host sorgente un **choke packet** contenente la destinazione del pacchetto, mentre quest'ultimo viene etichettato per impedire la generazione di altri choke packet. Il pacchetto

viene quindi inoltrato. L'host sorgente che riceve il choke packet deve ridurre il traffico inviato alla destinazione specificata.

In dettaglio, il router misura, per ciascuna linea, l'utilizzo istantaneo U e accumula, entro una media esponenziale M, la storia passata:

$$M_{nuovo} = a\, M_{vecchio} + (1 - a)U$$

dove

- il parametro a (compreso fra 0 ed 1) è il peso dato alla storia passata;
- (1 - a) è il peso dato all'informazione più recente.

Quando, per una delle linee in uscita, M si avvicina a una soglia di pericolo prefissata, il router esamina i pacchetti in ingresso per vedere se sono destinati alla linea d'uscita che è in allarme. In caso affermativo, invia all'host di origine del pacchetto un choke packet per avvertirlo di diminuire il flusso. Quando l'host sorgente riceve il choke packet diminuisce il flusso ed ignora i successivi choke packet per un tempo prefissato. Trascorso tale tempo prefissato, l'host si rimette in attesa di altri choke packet. Se ne arrivano altri, riduce ancora il flusso. Altrimenti, aumenta di nuovo il flusso.

3.2.2. HOP-BY-HOP CHOKE PACKET.

Su lunghe distanze ovvero ad alte velocità, la tecnica del choke packet risulta inefficiente a causa della lentezza di reazione; l'host che produce i pacchetti impiega del tempo a ricevere i choke packet ed a diminuire di conseguenza il ritmo della trasmissione.

Per migliorare è possibile utilizzare una variante del choke packet, chiamata ***hop-by-hop choke packet***. In tale approccio, ogni router che riceve tali pacchetti è costretto a rallentare la trasmissione sul percorso.

Questa tecnica permette di sbloccare rapidamente la congestione nel punto di creazione della stessa, ma richiede un maggiore utilizzo del buffer nei router sul percorso dall'host originario a quel router.

3.2.3. DROP-TAIL.

In caso di fallimento dei metodi precedentemente descritti, è possibile utilizzare tecniche operanti a livello di collegamento, che operano attraverso meccanismi brute-force, quali il drop-tail o il load shedding.
La politica più elementare utilizzata per la gestione delle code viene definita dal meccanismo denominato **Drop-Tail**, che opera a livello di collegamento attraverso la cancellazione dei pacchetti in coda una volta raggiunto il limite massimo del buffer (overflow); pertanto tale schema non reagisce dinamicamente alle condizioni di traffico della rete, e presenta inoltre diversi svantaggi:

- *problema della sincronizzazione dei flussi e della non equa distribuzione della perdita di pacchetti tra le connessioni:* la congestione in atto porta ad una cancellazione indiscriminata di tutti i pacchetti in arrivo, da cui scaturisce la non equa distribuzione della perdita tra le connessioni;
- *problema dello scarso utilizzo delle risorse di rete*: le connessioni che subiscono perdite di dati rispondono con un contemporaneo decremento della finestra, che a sua volta genera un calo di traffico in rete, e, di conseguenza, delle risorse a disposizione.

3.2.4. LOAD SHEDDING.

Il termine deriva dalla tecnica utilizzata per l'interruzione di energia elettrica da parte del gestore in caso superata disponibilità, ovvero l'interruzione secondo aree geografiche o importanza, per evitare il collasso dell'infrastruttura (caduta di carico). Un router, raggiunto il limite massimo del buffer, può decidere di scartare i pacchetti seguendo precise regole, e dipendenti dall'applicativo in esecuzione:

- regola del **wine** (vino); il vecchio è migliore del nuovo. In applicazioni quali il File Transfert è importante la sequenza dei pacchetti, e perciò è preferibile scartare i pacchetti nuovi. Tale regola può essere paragonata alle tecniche **F.I.F.O.** (First In, First Out).
- regola del **milk** (latte); il nuovo è migliore del vecchio. Nel caso di applicazioni Real Time, i pacchetti vecchi possono essere scartati. Si può parlare, in analogia, di **L.I.F.O.** (Last In, First Out).

L'implementazione di un criterio di eliminazione intelligente deve basarsi sull'assegnazione da parte delle applicazioni di un contrassegno di priorità sui pacchetti.

3.2.5. ACTIVE QUEUE MANAGEMENT (AQM).

Per superare almeno in parte i problemi legati al meccanismo Drop-Tail e Load Shedding, sono nati gli schemi **AQM, Active Queue Management** (Amministrazione attiva di Coda), algoritmi di controllo della congestione e anch'essi operano a livello di collegamento. Il principio che separa gli schemi AQM dal Drop-Tail è quello di comunicare la presenza di congestione preventivamente, in modo da causare il decremento dei dati spediti dalla sorgente prima che questi causino overflow e conseguente perdita di pacchetti.

L'AQM è perciò un metodo pre-reattivo che informa sulla presenza di congestione prima che si verifichi overflow nel buffer. Questo viene implementato tramite l'utilizzo di *Dynamic Buffer Limiting* (DBL). Il DBL tiene traccia della lunghezza della coda per ogni flusso di traffico entrante su singolo ingresso del router. Quando la lunghezza di coda di un flusso eccede il suo limite, il DBL comincia o a cancellare pacchetti o a impostare i bit *Explicit Congestion Notification* (ECN) nell'intestazione dei pacchetti IP a 1. ECN è descritto in RFC 3168.

I diversi algoritmi AQM differiscono nel tipo di parametri usati come indicatori di congestione, nella politica usata per riconoscere la congestione e nella politica usata per adattare la probabilità di cancellazione dei pacchetti in relazione alle condizioni di congestione della rete.

Uno dei più popolari schemi AQM utilizzati in reti TCP/IP viene chiamato **Random Early Detection, RED** (Indagine Random Anticipata). RED è un algoritmo che indaga sulla presenza di congestione nella rete tramite la valutazione della grandezza media della coda, calcolata con l'utilizzo di un filtro pesato, e reagisce con la cancellazione di pacchetti, in base ad una determinata probabilità, quando la grandezza è compresa tra un limite minimo e un limite massimo, certamente quando la grandezza eccede il limite massimo.

E' ora largamente accertato che una coda RED si comporti meglio di una Drop-Tail, anche se le prestazioni di una rete possono essere migliorate attraverso un oculato settaggio dei parametri.

Numerosi studi hanno infatti provato che in reti di diversa tipologia, il cambiamento dei parametri di RED può dar luogo prestazioni molto differenti.

3.2.6. RED - RANDOM EARLY DETECTION.

In questa sezione viene presentato l'algoritmo RED (Random Early Detection), proposto da Floyd e Jacobson nel 1993, per la prevenzione della congestione nelle reti a commutazione di pacchetto.

Il funzionamento principale di questo algoritmo è quello di rilevare la congestione tramite il controllo della lunghezza della coda. Una volta rilevata la congestione, il router può informare i suoi vicini della sua presenza utilizzando i bit ECN (Explicit Congestion Notification – notifica esplicita di congestione) nell'intestazione dei pacchetti.

Generalmente i router mantengono una lunghezza di coda relativamente bassa, per poi poter gestire dei periodi con traffico più consistente, che però non possono durare per un tempo troppo lungo.

L'algoritmo di RED non prevede una particolare modalità di notificazione della congestione: si può decidere arbitrariamente di optare per la cancellazione o per il semplice contrassegno dei pacchetti. Il contrassegno di un pacchetto ha la forma di un bit particolare dell'intestazione che viene settato ad uno, anche se questo dipende dal protocollo di trasporto utilizzato.

La cancellazione porta alla ritrasmissione dei dati, e ha quindi il difetto di aumentare il carico di traffico; mentre il contrassegno ritarda la risposta della finestra TCP che potrebbe ricevere abbastanza tardi la notifica o non riceverla affatto (in caso di perdita del pacchetto).

L'algoritmo di RED riscontra la congestione e misura il livello di carico di traffico usando la lunghezza media della coda (Average Queue Length).

Quando la lunghezza media è minore di una certa soglia minima t_{min} (minimum threshold, indicata anche con min_{th}), nessun pacchetto viene marcato o cancellato.

Quando, invece, essa eccede questa soglia minima, il router marca i pacchetti in maniera casuale con una probabilità data.

La probabilità che un pacchetto sia cancellato o contrassegnato (packet-marking probability) dipende:

- dal valore dell'Average Queue Length;
- dal tempo trascorso dall'ultima cancellazione;
- da un parametro: la probabilità massima di cancellazione (indicata con p_{max} o con max_p), che non deve mai essere superata.

Se la lunghezza media della coda è maggiore di una data soglia massima t_{max} (oppure max_{th}), tutti i pacchetti in arrivo vengono cancellati o contrassegnati, come mostrato in Figura 20.

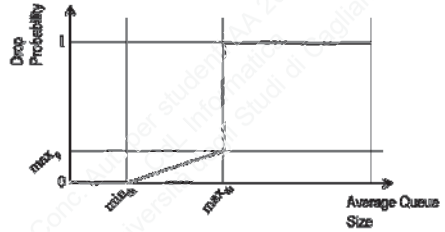

Figura 20. La probabilità di marcatura per la versione classica di RED.

L'algoritmo di RED viene proposto in due versioni: la prima di queste è quella classica, il cui calcolo della probabilità di marcatura è mostrato in Figura 20.
Nella *gentle-version*, invece, quando la lunghezza media della coda supera la soglia massima, la probabilità di marcatura non diventa unitaria, ma continua a crescere, anche se più velocemente rispetto a prima. Essa diventa unitaria solo dopo che la lunghezza media della coda supera 2 * max_{th}, come mostrato in Figura 21.

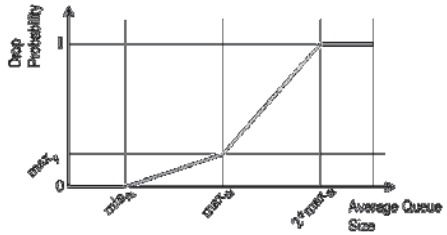

Figura 21. La probabilità di marcatura per la gentle-version di RED.

Si evince che, dato un livello di congestione, il router reagisce di conseguenza, cancellando o contrassegnando una quantità di pacchetti adattata dinamicamente. L'obiettivo è quello di marcare pacchetti in maniera abbastanza distanziata nel tempo in modo da evitare picchi di traffico, e in maniera sufficientemente frequente da evitare la crescita sconsiderata della lunghezza media della coda.
Il metodo è sicuramente brillante, ma nasconde un'insufficienza: è risaputo che la lunghezza media della coda cresce proporzionalmente al numero di connessioni attive nel sistema; questo si verifica fino a quando non viene raggiunta la soglia massima, dopo la quale tutti i pacchetti vengono cancellati o contrassegnati. Questo è un tentativo radicale di eliminare la congestione che può però sfociare in una sincronizzazione globale.
Con RED, solitamente, si decide di misurare la lunghezza della coda in byte, invece che in pacchetti. Con questa opzione la lunghezza media della coda riflette accuratamente il ritardo medio al router.
L'algoritmo di RED, inoltre, può essere modificato per assicurare che la packet-marking probability (probabilità di marcatura) di un pacchetto, sia proporzionale alla sua grandezza.

In questo caso, ad esempio, un grosso pacchetto FTP ha più probabilità di essere marcato rispetto ad un piccolo pacchetto TELNET. Questa logica tiene in conto la relazione tra dimensione del pacchetto e la congestione potenziale indotta.

Un altro degli aspetti caratterizzanti di RED è il non reagire alle congestioni temporanee; esse, infatti, sono accompagnate da un momentaneo aumento di lunghezza della coda (Queue Length). La congestione di lunga durata, invece, si riflette nell'aumento di lunghezza media della coda (Average Queue Length: la grandezza monitorata da RED) e porta, quindi, alla richiesta di decremento della finestra per alcune delle connessioni.

Anche in questo caso lo scopo è di privilegiare i carichi duraturi come fautori di congestione potenziale indotta, trascurando i soli picchi di traffico.

Nonostante i difetti che RED presenta, si può dire che la gestione dinamica e preventiva della congestione aumenta di molto le prestazioni della rete e risolve quelli che erano i più grandi problemi dell'algoritmo di *Drop-Tail*, garantendo:

- *L'evitare la congestione*. Con RED la rete viene costantemente monitorata e la sua politica di marcatura preventiva dei pacchetti porta ad una minore perdita di dati, con conseguente minore numero di pacchetti rispediti. La scelta della modalità di marcatura ottiene risultati migliori a seconda della situazione: in caso di cancellazione si ha un immediato alleviamento della congestione, ma una conseguente ritrasmissione dei dati; quando invece i pacchetti vengono contrassegnati, si deve fare più affidamento sulla "onestà" dei mittenti, cioè sul fatto che effettivamente decrementino la grandezza della loro finestra.
- *Tempi appropriati*. Dopo aver reso nota una congestione in seguito alla marcatura (tramite contrassegno) di un pacchetto, occorre almeno un Round Trip Time per vedere una diminuzione dell'Arrival Rate (la quantità di dati in arrivo). In RED la scala temporale per l'individuazione della congestione corrisponde approssimativamente alla scala temporale richiesta alla connessione per rispondere alla congestione.
- *Nessuna sincronizzazione globale*. L'intervallo di marcatura dei pacchetti dipende dal livello di congestione. Quando il livello di congestione è basso, il router ha una bassa probabilità di marcare ogni pacchetto che arriva, e quando aumenta il livello di congestione la probabilità di marcare pacchetti aumenta. I router RED evitano la sincronizzazione globale marcando meno pacchetti possibile. Inoltre RED elimina questo pericolo tramite la casualità della scelta del pacchetto da cancellare o contrassegnare.
- *Semplicità*. L'algoritmo di RED può essere implementato con un carico minimo nelle reti.
- *Massimizzazione del potere globale* (rapporto tra la velocità dei dati spediti ed il ritardo). In simulazioni con grande utilizzo della rete si notano prestazioni migliori di RED rispetto a quelle di *Drop-Tail*.
- *Gestione equa*. Un obiettivo del meccanismo per evitare la congestione è l'equità. Una gestione equa fa in modo da non discriminare nessuna connessione o classe di connessione. La quantità di pacchetti marcati per ogni connessione è proporzionale all'utilizzo che essa fa della larghezza di banda. Comunque, non si può assicurare che, effettivamente, ogni utente riceva lo stesso servizio, e d'altronde RED non si preoccupa neanche di controllare "l'onestà" degli utenti. L'algoritmo ha però dei meccanismi che permettono di identificare le connessioni che stanno utilizzando una grossa fetta della banda disponibile.
- *Appropriatezza per una gran quantità di ambienti*. Il meccanismo casuale di marcatura dei pacchetti è appropriato per reti con un grosso range di RTT diversi e larghezze di banda differenti, oltre ad un gran numero di connessioni attive allo stesso tempo. Inoltre, cambiamenti nel carico

portano a cambiamenti nella lunghezza media della coda e quindi la probabilità di cancellazione viene calcolata di conseguenza.

ALGORITMO DI RED.

L'algoritmo che calcola il valore medio della lunghezza della coda determina il grado massimo di trasmissione che il router può supportare. L'algoritmo che calcola la probabilità di marcare un pacchetto determina quanto frequentemente il router marca i pacchetti in relazione al livello di congestione. Tramite una cancellazione di pacchetti a determinati intervalli di tempo si tiene sotto controllo la lunghezza media della coda.
L'algoritmo dettagliato di funzionamento dei router RED è il seguente.

Inizializzazione:

$avg = 0$
$count = -1$

Per ogni pacchetto in arrivo calcola la nuova lunghezza media della coda avg:
    se la coda non è vuota
        $avg = (1 - w_q)avg + w_q q$
    altrimenti
        $m = f(time - q\_time)$
        $avg = (1 - w_q)^m avg$
    se $min_{th} \leq avg \leq max_{th}$
        incrementa $count$
        calcola la probabilità p_a:
            $p_b = max_p(avg - min_{th})/(max_{th} - min_{th})$
            $p_a = p_b/(1 - count \cdot p_b)$
        con probabilità p_a:

            marca il pacchetto in arrivo
            $count = 0$
        altrimenti se $max_{th} \leq avg$
            marca il pacchetto in arrivo
            $count = 0$
        altrimenti $count = -1$
Quando la coda diventa vuota
    $q\_time = time$

In questo algoritmo possiamo distinguere: avg è la lunghezza media della coda, q_time è il momento di inizio dello stato di idle della coda (cioè il periodo in cui la coda rimane vuota), $count$ è il numero di pacchetti instradati senza problemi prima che venisse marcato l'attuale pacchetto, w_q è il peso della coda, min_{th} è il limite minimo della coda, max_{th} è il limite massimo della coda, max_p è il valore massimo di p_b, p_a è la probabilità che l'attuale pacchetto venga marcato, q è la lunghezza attuale della coda, $time$ è l'istante attuale e $f(t)$ è una funzione lineare del tempo. Il calcolo di questo algoritmo prende in considerazione il periodo di idle della coda, e calcola il numero di pacchetti che potrebbero essere messi in coda durante quel periodo, come se fossero realmente arrivati al router.
In alternativa a questa implementazione dell'algoritmo RED, si può considerare la grandezza della coda in byte piuttosto che in pacchetti.
Con tale modifica, l'algoritmo può essere riespresso in questo modo:

$p_b = max_p(avg - min_{th})/(max_{th} - min_{th})$
$p_b = p_b$PacketSize/MaximumPacketSize
$p_a = p_b/(1 - count \cdot p_b)$

In questo caso verrà marcato con più probabilità un pacchetto grande piuttosto che uno piccolo.

Il peso della coda w_q è determinato dalla grandezza della coda e dalla durata del periodo con un traffico più consistente del normale.

CALCOLO DELLA LUNGHEZZA MEDIA DELLA CODA.

I router RED utilizzano un filtro passa-basso per il calcolo della lunghezza media della coda. Anche se si verifica un inizio di congestione o un aumento improvviso del traffico, la lunghezza media della coda non varia considerevolmente, coerentemente con quanto detto sul privilegiare carichi duraturi come fattori di congestione potenziale indotta.
Il filtro passa-basso è una media dinamica pesata esponenziale (*EWMA – Exponential Weighted Moving Average*):

$$avg = (1 - w_q)avg + w_q q$$

Il peso w_q è una costante del tempo del filtro passa-basso.

LIMITE SUPERIORE PER W_Q.

Se w_q è troppo grande, il metodo del calcolo della media può non filtrare la congestione iniziale.
Si supponga che inizialmente la coda sia vuota e quindi la sua lunghezza media sia zero, e che successivamente la coda cresca da 0 sino ad L, dove L è il numero di pacchetti arrivati. Perciò all'arrivo dell'L-esimo pacchetto la lunghezza media della coda avg_L è:

$$avg_L = \sum_{i=1}^{L} i w_q (1-w_q)^{L-i} = w_q (1-w_q)^L \sum_{i=1}^{L} i \left(\frac{1}{1-w_q}\right)^i = L + 1 + \frac{(1-w_q)^{L+1} - 1}{w_q}.$$

Dato un limite minimo min_{th}, e considerato che si vuole riuscire a gestire un aumento di carico che arrivi fino a L pacchetti in arrivo al router, allora w_q dovrebbe essere scelto in modo da avere che $avg_L < min_{th}$, cioè:

$$L + 1 + \frac{(1-w_q)^{L+1}}{w_q} < \min_{th}$$

LIMITE INFERIORE PER W_Q.

E' ragionevole scegliere un buon limite inferiore per w_q, in quanto non è desiderabile che la lunghezza media della coda *avg* non reagisca troppo rapidamente e che il router non riesca a rendersi conto della congestione nascente.
Si consideri che in un primo momento la coda passi da essere vuota a contenere un pacchetto, e che poi i pacchetti arrivino e vengano rispediti alla stessa velocità, perciò che nella coda rimanga sempre un pacchetto. Inoltre si consideri che inizialmente la lunghezza media della coda sia zero. In questo caso la coda prende $-1/\ln(1-w_q)$ pacchetti in arrivo (con la lunghezza della coda che rimane a uno) sino a che la lunghezza media della coda giunge a $0.63 = 1 - 1/e$. Nella maggior parte dei simulatori si utilizza $w_q = 0.002$.

ASSEGNAZIONE DI MIN_{TH} E MAX_{TH}.

I valori dei limiti minimo e massimo, min_{th} e max_{th}, dipendono dalla lunghezza media della coda che si vuole mantenere. Se ad esempio si vuole mantenere un traffico molto intenso per abbastanza tempo, il limite minimo deve essere abbastanza alto, come anche quando si vuole che il ritardo di un pacchetto non influenzi eccessivamente l'affidabilità del collegamento.
Per quanto riguarda il limite massimo, questo dipende largamente dal massimo ritardo medio che il router è disposto ad accettare.

I router RED lavorano meglio quanto la differenza $max_{th} - min_{th}$ è maggiore del tipico aumento della lunghezza media della coda in un RTT (*Round Trip Time* – tempo di andata e ritorno). Una regola che quasi sempre è adatta è quella di scegliere un max_{th} uguale al doppio del min_{th}.

CALCOLO DELLA PROBABILITÀ DI MARCARE UN PACCHETTO.

La probabilità iniziale di marcare un pacchetto p_b è calcolata come una funzione lineare della lunghezza media della coda, con la seguente formula:

$$p_b = max_p(avg - min_{th})/(max_{th} - min_{th}).$$

dove max_p rappresenta il valore massimo che la probabilità di marcare un pacchetto p_b può assumere.

Un metodo per calcolare la probabilità finale di marcare un pacchetto utilizza una variabile geometrica random. Assumiamo che X sia il tempo che intercorre tra due pacchetti che vengono marcati, ossia il numero di pacchetti che arrivano dopo un pacchetto marcato senza essere a loro volta marcati. Dato che ogni pacchetto viene marcato con una probabilità p_b, allora:

$$Prob[X = n] = (1 - p_b)^{n-1} p_b.$$

Dato che X è una variabile geometrica standard di parametro p_b, avrà una media $E(X) = 1/p_b$.

Avendo una lunghezza media di coda costante, è auspicabile che non vi sia un intervallo né troppo breve né troppo grande tra due pacchetti marcati. Entrambi questi effetti possono portare alla sincronizzazione globale, cioè la situazione che si presenta quando molte connessioni riducono la grandezza della propria finestra contemporaneamente, che è appunto ciò che accade con l'utilizzo di questo primo metodo di calcolo della probabilità.

Un altro approccio è quello di utilizzare una variabile uniforme random X, che assuma i valori $\{1, 2, \ldots, 1/p_b\}$, assumendo per semplicità che p_b sia un intero. Per ottenere ciò, la probabilità di marcare un pacchetto deve essere $p_b/(1 - count \cdot p_b)$, dove *count* è il numero di pacchetti non marcati che sono arrivati dopo l'ultimo pacchetto marcato. In questo caso si ha che:

$$Prob[X = n] = \frac{p_b}{1-(n-1)p_b} \prod_{i=0}^{n-2} \left(1 - \frac{p_b}{1-ip_b}\right)$$

$$= p_b \text{ con } 1 \le n \le 1/p_b,$$

$$Prob[X = n] = 0 \text{ con } n > 1/p_b.$$

Con questo secondo metodo la media $E[X] = 1/(2p_b) + 1/2$.

IMPLEMENTAZIONE DELL'ALGORITMO RED.

L'implementazione da parte dei router dello schema RED porta all'attuazione di alcuni degli scopi dell'algoritmo RED stesso.

La **prevenzione della congestione** si attua grazie alla cancellazione di alcuni pacchetti quando viene raggiunto il limite massimo. In questo modo il calcolo della lunghezza media della coda non supererà il limite massimo.

Se viene assegnato un giusto valore a w_q per la procedura descritta, il router controllerà la reale lunghezza media della coda. Se invece di cancellare il pacchetto, quando viene raggiunto il limite massimo, vengono impostati i bit dell'intestazione, si renderà possibile una cooperazione nella prevenzione della congestione da parte degli altri router.

Un'**appropriata bilancia temporale** permette che, dopo che si notifica una congestione con il marcamento del pacchetto, in un tempo pari al Round Trip Time (RTT), il router noti una diminuzione di velocità di arrivo dei pacchetti.

La bilancia temporale, quindi, misura il tempo di reazione alla congestione delle connessioni.

La **sincronizzazione globale** è da evitare, e per fare questo si devono marcare il minor numero di pacchetti possibile se la congestione è bassa, tramite il calcolo della probabilità di marcare un pacchetto. Questa infatti è più bassa quanto più è bassa la congestione.

In questo modo non si avrà un crollo del traffico improvviso da parte di tutti le connessioni.

E' necessario che gli algoritmi dei router RED siano implementati con una spesa abbastanza ridotta per le reti attuali, quindi l'implementazione stessa deve essere semplice.

La **massimizzazione del rapporto tra larghezza di banda e ritardo globale** si ottiene tramite il controllo esplicito della lunghezza media della coda. Infatti è dimostrato che il suddetto rapporto è più alto con l'utilizzo dei gateway RED piuttosto che di quelli con normale Drop Tail.

Uno degli scopi della prevenzione della congestione è il **corretto bilanciamento** nella condivisione della coda. I router RED non devono discriminare certe connessioni rispetto alle altre, infatti il numero di pacchetti marcati di una certa connessione sarà proporzionale alla larghezza di banda di tale connessione.

I router RED non controllano che ogni connessione sfrutti uguale larghezza di banda, ma è un controllo che si può aggiungere. In particolare, un controllo del genere è utile proprio durante un periodo di congestione.

I router RED devono essere **adatti ad un grande insieme di ambienti**. Il meccanismo di scelta dei pacchetti da marcare fa si che si marchi con più probabilità un pacchetto di una connessione che sfrutta più larghezza di banda o con diversi valori di *RTT*, o quello di un host che sfrutta più connessioni contemporaneamente.

Cambiamenti nel traffico vengono notati dal cambiamento della lunghezza media della coda, e la velocità con cui i pacchetti vengono marcati è proporzionale alla quantità di congestione.

3.2.7. PI - PROPORTIONAL INTEGRAL.

PI (Proportional Integral) viene formulato come evoluzione di RED, e si propone di risolvere quelli che sono i problemi che maggiormente affliggono il primo degli schemi AQM.

RED, infatti, mostra un legame eccessivo e dannoso tra la lunghezza media della coda e la probabilità di marcatura dei pacchetti.

Inoltre, a seconda di come vengono settati i parametri e a seconda del tipo di rete, si hanno prestazioni notevolmente differenti. RED può essere quindi considerato eccessivamente sensibile al cambiamento dei parametri e produce un errore detto *steady state*.

Con l'introduzione del controller PI, si riesce a portare l'errore di steady state a zero. E' infatti risaputo che la lunghezza della coda risulta essere molto più stabile con PI.

Le caratteristiche principali di questo algoritmo sono le seguenti:

- PI cerca di mantenere una data lunghezza della coda considerata campione.
- PI campiona la lunghezza istantanea della coda ad intervalli precisi e calcola la probabilità di cancellazione al *k*-esimo campione:

Prima del PI controller, nasceva il Proportional controller. I vantaggi del primo sul secondo non sono gratuiti. Sebbene l'errore dello *steady state* venga eliminato, la velocità di risposta diminuisce notevolmente. Il che si traduce in lunghi ritardi di incodamento e grosse quantità di pacchetti persi.

Un altro problema di PI è la cattiva risposta alle oscillazioni di traffico nella rete, sia nei momenti in cui il numero degli utenti decresce improvvisamente, sia in quelli in cui il numero dei flussi cresce molto rapidamente (la probabilità di marcatura dei pacchetti, infatti, aumenta lentamente).
Per questo è stato proposto un ulteriore algoritmo P^2I che risolve il problema della risposta lenta di PI. P^2I è una combinazione di PI e Proportional controller.

bianca

CAPITOLO 4.

SCHEMI AQM:

ESPERIENZE SIMULATIVE.

4.1. INTRODUZIONE.

In oltre dieci anni sono stati proposti numerosi schemi Active Queue Management (AQM) in letteratura. Molti di questi studi sono stati condotti alla volta di migliorare il controllo della congestione nelle reti best-effort. Talvolta, si è andati incontro ad una ridotta analisi di valutazioni standard delle performance degli schemi AQM. Uno studio rigoroso sulla parametrizzazione di schemi specifici, e l'istituzione di un criterio di confronto comune, è essenziale per valutazioni oggettive di schemi differenti.

Diversi schemi AQM sono apparsi in letterature dall'originale **Random Early Detection (RED)** proposto da Floyd e Jacobson nel 1993. Ciò nonostante, AQM non è ancora largamente utilizzato in Internet e la maggior parte dei router sono basati sulle semplici code Drop-Tail.

La lentezza con cui AQM viene sviluppato è dovuta perlopiù alla mancanza di dettagliati schemi di valutazione oggettiva.

Sono stati portati a termine alcuni studi di confronto su un numero limitato di schemi AQM che coinvolgono sottoinsiemi coerenti di parametri. I risultati differenti possono essere ottenuti da parametrizzazioni funzionalmente identiche

di alto livello, perché vengono utilizzati differenti modelli di base o diverse tecniche di simulazione.

E' stato proposto un framework integrato per testare gli schemi AQM, che fornisce una sofisticata capacità nella valutazione di schemi AQM usando una varietà di scenari di simulazione. Tale framework fornisce un interfaccia intuitiva e robusta. Per ottenere risultati statisticamente corretti il framework calcola il numero di simulazioni richieste, assieme al tempo necessario per ogni simulazione, dinamicamente.

4.2. METRICHE DI VALUTAZIONE.

Il nucleo del framework è formato da un piccolo numero di metriche per la valutazione delle performance della rete, accuratamente scelte.

Queste:

- sono **rappresentative dell'esperienza reale dell'utente finale** nelle performance della rete. Ad esempio, viene considerato il jitter end-to-end piuttosto che la stabilità della lunghezza della coda. Se le metriche router-based sono rilevanti per capire il comportamento di un dato schema, così il confronto dovrebbe essere basato sull'esperienza dell'utente finale.
- **concentrano un ampio range di risultati di performance** in un piccolo numero di metriche significative; quindi si riduce il sovraccarico computazionale e si facilita il significativo confronto di schemi.
- **riflettono le performance dell'intera rete**, cioè non dipendono dalle misurazioni prese in un singolo punto della rete, quindi sono indipendenti dallo scenario di rete che viene considerato, conferendo alla metrica di valutazione una valenza estesa sull'intera sottorete considerata.

Vengono scelte a tale scopo cinque metriche, che rappresentano: l'utilizzazione della rete, il ritardo, il jitter, il tasso di cancellazione di pacchetti e l'"onestà" nella condivisione della rete.

- **Utilisation metric.** Questa è definita come la percentuale della capacità totale della rete utilizzata durante il ciclo di simulazione. Per una determinata combinazione di traffico e topologia, la capacità della rete è definita come il massimo flusso totale nella rete. In tal modo si fornisce un limite massimo sulla quantità totale di dati ricevuti ottenibile. Per calcolare la metrica da cui si ottiene l'utilizzazione totale, la somma dei dati ricevuti dall'insieme di tutti i flussi deve essere divisa con la capacità della rete calcolata.
- **Delay metric.** La delay metric descrive la media del ritardo end-to-end sperimentato dai pacchetti. Prima si calcola la media del ritardo di tutti i pacchetti in un flusso singolo, e successivamente dalle medie ottenute si quantifica quella su tutti i flussi.
- **Jitter metric.** Questa metrica si basa sulla varianza del ritardo complessivo dei pacchetti. Prima viene calcolato il coefficiente di variazione del ritardo per ogni flusso, poi viene valutata la media di questi valori per tutti i flussi.
- **Drop metric.** Questa metrica rappresenta la percentuale di pacchetti cancellati. Include sia i pacchetti cancellati a causa della politica AQM (cancellazioni anticipate) sia quelli cancellati a causa di overflow del buffer (cancellazioni per superamento limite finestra). Non viene fatta differenza tra questi in quanto hanno lo stesso effetto sulla percezione dell'utente finale.
- **Fairness metric.** Il **corretto bilanciamento** nella condivisione della capacità della rete per ogni singolo flusso è calcolata staticamente facendo riferimento alla capacità della rete, usando la metrica di onestà di Jain et al. Se ogni flusso mantiene la sua condivisione della capacità della rete bilanciata, il valore della metrica è 1; questo valore decresce se le risorse sono condivise irregolarmente dai flussi.

4.3. SCENARI DI SIMULAZIONE PER LA VALUTAZIONE AQM.

Per poter definire una corretta ipotesi di lavoro, uno dei fattori critici nella strutturazione di un framework di valutazione è la scelta dell'ambiente di simulazione. Uno scenario di rete può essere definito tramite due elementi: una topologia di rete statica e una combinazione di traffico che fluisce tra i nodi della rete.

La topologia e la combinazione di traffico offrono un ampio insieme di parametri da modellare. Il modello scelto a titolo esemplificativo per l'ottenimento di un significativo risultato, e di performance AQM affidabili, incorporerà caratteristiche peculiari di Internet.

4.3.1. TOPOLOGIA DELLA RETE.

Gli effetti di differenti algoritmi di gestione delle code sulle performance in uno scenario con un singolo router può essere valutato utilizzando la topologia di Dumbbell, in Figura 22.

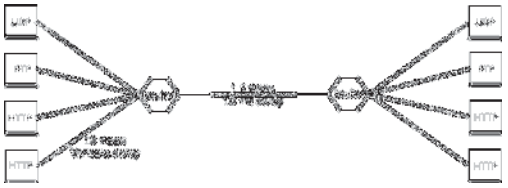

Figura 22.Topologia di Dumbbell.

Questa configurazione unisce la semplicità del modello ed è largamente utilizzata; ciò consente di operare con il framework facilmente senza ledere la generalità della trattazione.

Per definire la topologia di Dumbbell vengono utilizzati tre parametri: la larghezza di banda del collegamento a collo di bottiglia, il suo ritardo di propagazione, e l'intervallo di round-trip time (RTT) di ogni flusso. Quest'ultimo è uno dei parametri frequentemente trascurati negli scenari comuni di simulazione. Studi di misurazione hanno mostrato che gli RTT rilevati in Internet possono variare profondamente, con la maggior parte di RTT compresi in un intervallo tra 15 e 500 ms.

E' utile ricordare che RTT varia significativamente a seconda della tratta reale impegnata (dorsale, primaria, secondaria), e ha diversi livelli di tolleranza proprio in funzione di quest'ultime.

4.3.2. COMPOSIZIONE DEL TRAFFICO.

La creazione di un modello realistico di traffico in Internet necessita dell'uso di un largo insieme di parametri. Questi parametri si ottengono da studi di misurazioni del traffico di Internet. Una tipica composizione di traffico utilizzata per la valutazione di algoritmi di amministrazione di code nei router è formata

da trasferimento di massa tramite flussi FTP di lunga durata. Questa composizione non è rappresentativa del traffico dell'Internet attuale: questa è dominata da un'alta porzione di flussi Web a vita breve, detti anche **libellule**, e un basso numero di trasferimenti di lunga vita, detti anche **tartarughe**.

Inoltre, non dovrebbe essere trascurato l'effetto che produce il traffico che non utilizza controllo della congestione end-to-end nella realizzazione della rete e nella cessione di larghezza di banda.

Il framework di valutazione prevede l'inclusione di una certa quantità di traffico UDP in ogni scenario di simulazione, anche se la composizione del traffico di default è principalmente Web con uno sfondo di flussi FTP e UDP.

4.3.3. IMPLEMENTAZIONE DEL FRAMEWORK

L'architettura del framework è strutturata in due componenti: un'interfaccia di alto livello per specificare scenari di simulazione ed esperimenti, e un motore per eseguire molte simulazioni in parallelo.

INTERFACCIA DI ALTO LIVELLO.

NS2 è lo standard *de-facto* dei software simulatori di reti di trasmissione dati. Un framework di valutazione dovrà consentire di ottenere risultati di simulazione con l'applicazione di semplici parametri di configurazione con script non molto complessi. E' utile in tale ipotesi tenere conto degli obiettivi perseguiti riconducibili alla verificabilità:

- dell'effetto dell'aumento di traffico non affidabile UDP sulla realizzazione della sottorete e sulla condivisione di larghezza di banda;
- dell'affidabilità degli schemi Active Queue Management per descrivere il funzionamento con differenti carichi di traffico WEB;

– dell'influenza indotta dal settaggio dei parametri nella realizzazione di un determinato AQM scheme.

La configurazione e l'esecuzione di simulazioni che rispondono a queste caratteristiche, utilizzando NS2, rappresentano un processo time-consuming. Il processo prevede la creazione di scenari di simulazione in NS2, eseguendo numerose simulazioni ed elaborando l'output ottenuto da NS2 per derivarne i risultati.

Tali processi sono di norma agevolati dalla disponibilità di script all'uopo predisposti. L'elaborazione e l'analisi dei dati di output rappresenta una fase successiva per l'ottenimento di risultati credibili di simulazione.

Nel framework, NS2 si comporterà come "motore di simulazione".

Il sistema è stato inizialmente previsto per la valutazione di AQM, ma può ugualmente essere utilizzato per rispondere ad altre domande per la valutazione nella realizzazione di una rete.

Il framework dovrà consentire di operare con un numero di simulazioni che hanno in comune il valore fisso di determinati parametri ed il valore variabile di uno dei parametri. I parametri che rimangono fissi potranno essere sia valori specificati dall'utente che predefiniti di default. Tali esperimenti forniranno delle funzionalità per l'esecuzione di simulazioni associate, e calcoleranno le metriche di funzionamento della rete usando i dati in output. Per ogni modello di simulazione viene eseguito un certo numero di repliche con flussi indipendenti di numeri casuali. Dai valori delle metriche ottenuti nelle diverse repliche si stimerà un valore significativo, e verrà riportato il livello di errore relativo ai valori delle metriche stesse.

Una relazione conclusiva sarà il documento riassuntivo dei risultati ottenuti per gli intervalli di valori desiderati.

MOTORE PER SIMULAZIONI PARALLELE

Il framework di valutazione può richiedere molte simulazioni per ogni insieme di risultati che si vogliono ottenere. E' comune trovare scenari di simulazione NS2 che richiedono molti minuti per potersi considerare conclusi, utilizzando le tecnologie delle workstation attuali. La produzione di una relazione conclusiva richiede che venga compiuto un largo numero di simulazioni. Di conseguenza, utilizzare una singola workstation per generare le relazioni conclusive rischia di impiegare molto tempo.

Certi simulatori di rete sequenziali sono stati estesi per essere eseguiti su computer paralleli. L'approccio del framework differisce in quanto si ottengono economie di scala dalla computazione in parallelo di ogni esperimento conservando la granularità di una singola simulazione. L'utilizzo di questo framework coinvolge tipicamente l'esecuzione di molte simulazioni per ogni esperimento. I vantaggi di una velocità sostanziale possono essere ottenuti dalla distribuzione del carico tra un insieme di workstation connesse in rete. Simulazioni individuali vengono ancora eseguite utilizzando un simulatore standard NS2 sequenziale quindi fornendo una buona scalabilità senza il bisogno di modificare il software. La Figura 23 illustra l'architettura del framework quando si utilizza il motore di simulazione parallela con due host **slave**. Gli slave eseguono le simulazioni e interpretano le tracce di output. Il processo di controllo comunica con gli slave utilizzando XML-RPC, un protocollo di chiamata a procedura remota basata su HTTP e XML. Il controller manda i parametri dello scenario con le richieste, e gli slave rispondono con un sommario dei risultati della simulazione. Una volta che tutte le simulazioni sono finite e i loro risultati sono stati raccolti, il framework calcola le metriche di realizzazione della rete a livello sperimentale.

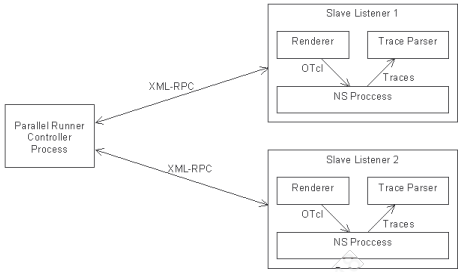

Figura 23. Architettura del motore per simulazioni parallele.

Non si trascurino le potenzialità offerte dall'estensione di questa architettura per simulazioni di particolare gravosità, fino ad operare su sistemi distribuiti di tipo grid.

4.3.4. PERFEZIONAMENTO DELL'AFFIDABILITÀ DEL FRAMEWORK.

Con la crescita in grandezza e complessità delle reti reali, la simulazione sta diventando lo strumento più importante usato per la valutazione della struttura della rete. Tuttavia non sempre esiste un'analisi corretta dei dati di output della simulazione.

In questa sezione vengono descritte le tecniche statistiche e le procedure implementate nel framework. Queste possono essere usate per ottenere risultati significativi dagli studi della struttura delle reti che si fondano su risultati ottenuti in simulazione.

ANALISI DI UNA SIMULAZIONE CON UN SOLO CICLO.

Il framework dovrebbe ottenere valori di stato costante per le cinque metriche. Per un singolo ciclo della simulazione, è necessario fare due considerazioni:

- La stima del valore dello stato costante è alterato da uno stato iniziale atipico, in quanto allo stato iniziale le code e i collegamenti sono vuoti (condizione transiente iniziale).
- Una volta che la fase iniziale, transitoria, è terminata, è necessario determinare quanto dovrebbe essere lungo il ciclo di simulazione in modo da ottenere una buona approssimazione del significativo stato costante (durata del ciclo di simulazione).

Possibili metodi per superare questi problemi vengono ora discussi.

CONDIZIONE TRANSIENTE INIZIALE.

Il modello di rete utilizzato nella simulazione è vuoto e inattivo all'inizio; tutte le code sono vuote e il traffico non attraversa la rete.

Queste condizioni iniziali solitamente introducono un errore in quanto non sono rappresentative dello stato costante desiderato.

Per calcolare la lunghezza del periodo iniziale transiente, è stata utilizzata una semplice formula sviluppata da White. Questo metodo, chiamato **Marginal Confidence Rule** (MCR - Regola di confidenza Marginale), seleziona il punto di troncamento che minimizza l'ampiezza dell'intervallo di confidenza, in relazione alla media del campione di troncamento. Se $x_1, x_2, \ldots, x_N$ è l'insieme dei punti misurati durante il tempo della simulazione, il punto di troncamento ottimale $i \in [1, N]$ è definito come:

$$i = avg \min_{0 \le i \le N} \left[\frac{z_{\frac{\alpha}{2}} \sigma_i(X)}{\sqrt{N-i}} \right], \quad (1)$$

dove $z_{\frac{\alpha}{2}}$ è il valore della distribuzione normale delle componenti con un intervallo di confidenza uguale a 100(1 - α)%, e $\sigma_i(X)$ è la deviazione standard del campione della sequenza $x_{i+1},\ldots,x_N$.

La confidenza α è fissa; per cui $z_{\frac{\alpha}{2}}$ è costante e l'equazione (1) può essere scritta come:

$$i = avg \min_{0 \le i \le N} \left[\frac{\sigma_i^2(X)}{N-i} \right]. \quad (2)$$

La Figura 24 mostra come la convergenza al valore dello stato costante è garantita dall'uso di MCR per rimuovere il valore iniziale errato.

La media del campione calcolata senza l'errore iniziale converge dopo 10 secondi, ma, nonostante questo, dopo 40 secondi di simulazione, l'effetto dell'errore iniziale sulla media del campione è ancora evidente.

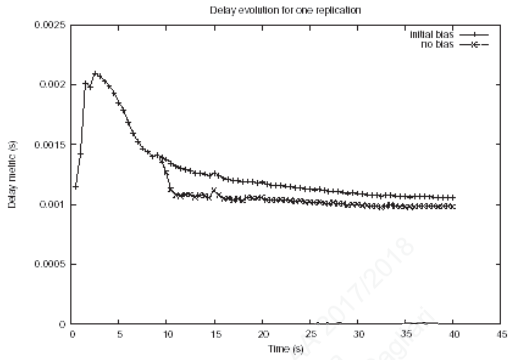

Figura 24. Troncamento MCR dell'errore iniziale.

Questo metodo è applicato alle cinque differenti metriche di interesse e, come si può vedere nella Figura 25, l'errore introdotto dal periodo iniziale passeggero è differente per ognuna di esse.

Il punto di troncamento utilizzato è il valore più alto dei cinque punti di troncamento.

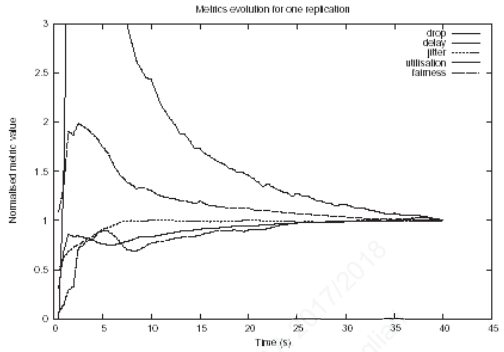

Figura 25. Convergenza delle cinque metriche, con un periodo iniziale passeggero. (medie del campione normalizzate al loro valore a t = 40s)

DURATA DEL CICLO DI SIMULAZIONE.

Per utilizzare il metodo di replica/cancellazione (descritto in seguito), Law e Kelton raccomandano un tempo di simulazione più grande della lunghezza del periodo iniziale passeggero, per essere sicuri che siano stati raccolti sufficienti dati. Nell'implementazione descritta, il punto di troncamento è calcolato ogni *k* punti, per evitare di utilizzare il calcolo di MCR (Marginal Confidence Rule), che può essere pesante, ripetuto ad ogni punto. La simulazione viene fermata quando il punto di troncamento *i* è per esempio $N > p \times i$ (dove *p* vale 5 in questo framework).

CICLI DI SIMULAZIONE MULTIPLI.

Molti studi su simulazioni di reti costruiscono un modello, selezionano metriche di realizzazione, e concludono con un ciclo singolo di simulazione per ottenere risultati. L'utilizzo di valori casuali scelti da distribuzioni particolari come valori di input per la simulazione, si riflettono nel fatto che i risultati ottenuti da un singolo ciclo di simulazione sono esattamente la realizzazione di variabili casuali. La figura 26 illustra come i valori in un punto, ottenuti per una determinata metrica di realizzazione, possono statisticamente variare considerevolmente tra cicli indipendenti di simulazione. Di conseguenza, è essenziale utilizzare repliche multiple indipendenti dello stesso modello di simulazione con analisi statistiche dei dati di output appropriate. In questo modo si diminuisce la probabilità di ottenere risultati errati e si migliora la credibilità degli stessi.

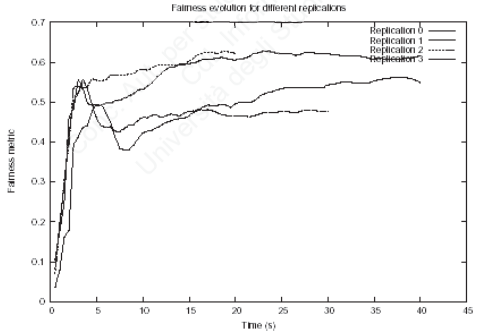

Figura 26. Valutazione della Fairness metric su un tempo di simulazione dato per differenti repliche della simulazione.

Quando si utilizzano repliche multiple indipendenti, il numero di repliche n può essere stabilito in anticipo, utilizzando il metodo di calcolo della **procedura della dimensione campione fissata**.

Quando n è fissata, possono essere ottenuti il punto di stima e l'intervallo di confidenza per la media $\mu = E(X)$, dove X rappresenta i valori che la metrica di interesse assume nelle diverse repliche.

Un valore $\overline{X}(n)$ in un punto può essere utilizzato per stimare la media μ, e di conseguenza un intervallo di confidenza $100(1 - \alpha)\%$ per μ può essere trovato con:

$$\delta(n,\alpha) = \overline{X}(n) \pm t_{n-1,1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}, \qquad (3)$$

dove $\overline{X}(n)$ è la media stimata, $S^2(n)$ è la varianza campione e $t_{(n,\alpha)}$ è la distribuzione t di Student. Lo svantaggio principale del calcolo della procedura della dimensione campione fissata è che questa non permette di calcolare la mezza lunghezza dell'intervallo di confidenza predefinito. Se un errore specifico o una precisione sono desiderati per la stima della media $\overline{X}(n)$, il numero di repliche n viene calcolato dinamicamente.

Il framework implementa un metodo per la decisione dinamica del numero di repliche richiesto.

La procedura utilizzata ha lo scopo di essere capace di ottenere una stima della media $\overline{X}$ per μ, con un errore relativo γ ($0 < \gamma < 1$) e una confidenza di $100(1 - \alpha)\%$, dove α e γ sono determinate inizialmente. L'errore relativo è definito come $\gamma = |\overline{X} - \mu|/|\mu|$, di conseguenza l'errore percentuale in $\overline{X}$ è $100\ \gamma\%$. Una procedura sequenziale per determinare il numero di repliche n è stato implementato nel framework.

Questo inizia con un dato numero di repliche $n_0 = 10$, imposta $n = n_0$ e calcola la stima della media $\overline{X}(n)$ e la metà della lunghezza dell'intervallo di confidenza $\delta(n,\alpha)$ ottenuta con la formula (3).

Se $\delta(n,\alpha)/|\overline{X}(n)| \leq \gamma$', l'errore relativo, è al di sotto della soglia desiderata, il valore corrente di $\overline{X}$ è utilizzato come il punto stimato per il valore medio della metrica μ e non si aggiungono ulteriori repliche.

L'intervallo corrente di confidenza $100(1 - \alpha)\%$ approssimato è dato da:

$$I(\alpha,\gamma) = [\overline{X}(n) - \delta(n,\alpha), \overline{X}(n) + \delta(n,\alpha)]. \qquad (4)$$

Il numero di repliche n è incrementato di 1 sino a che il valore di n viene raggiunto per cui $\delta(n,\alpha)$ è al di sotto della soglia predefinita. La Figura 27 mostra come i valori della metrica si sviluppano dipendentemente della lunghezza del ciclo di simulazione, una volta che l'errore introdotto dal periodo iniziale di warm-up è stato eliminato.

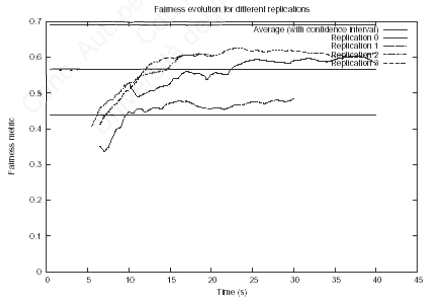

Figura 27. Valore della fairness metric in funzione del tempo senza l'errore iniziale, e media stimata con intervallo di confidenza.

La media stimata per la fairness metric con il suo intervallo di confidenza associato è ugualmente rappresentato nel grafico.
Quando si confronta questo grafico con quello in Figura 26 si può vedere come la cancellazione del periodo iniziale passeggero migliora la stabilità dei valori della fairness metric.
La procedura sequenziale per determinare il numero di repliche, quando utilizzata in unione con la tecnica descritta nella sezione precedente per eliminare l'errore iniziale passeggero, è conosciuta come **metodo di replica/cancellazione**.

4.3.5. STUDIO DEL CORRETTO BILANCIAMENTO DI RED.

Esistono delle varianti dell'algoritmo RED, come RED-PD (RED Preferential Dropping - RED con cancellazione preferenziale), che di seguito vengono rapidamente illustrate.
Anche la coda drop-tail è presa in considerazione dato che è attualmente lo schema base implementato nei router in Internet.
Gli algoritmi considerati sono:

- **Drop-Tail**: code drop-tail standard. I pacchetti vengono cancellati solamente quando avviene overflow del buffer.
- **RED**: la versione attuale di RED, implementata in NS2. Include la versione gentile (senza discontinuità nella funzione di probabilità di cancellazione), e miglioramento del setting dei parametri.
- **RED-PD**: RED con cancellazione preferenziale, aggiunge un ulteriore livello a RED, analizzando e penalizzando i flussi non affidabili.

L'esperimento si basa su un semplice scenario basato sulla topologia di Dumbbell. I parametri che vengono utilizzati sono:

- Collo di bottiglia con larghezza di banda a 10 *Mbps;*
- Collo di bottiglia con ritardo di 10*ms;*
- Intervallo di RTT compreso tra 20*ms* - 460*ms.*

Questo esperimento esamina gli effetti della proporzione di traffico non affidabile (UDP in questo caso) sulla realizzazione.
Si mettono in evidenza, in questo momento, solo le metriche di cancellazione e condivisione onesta.

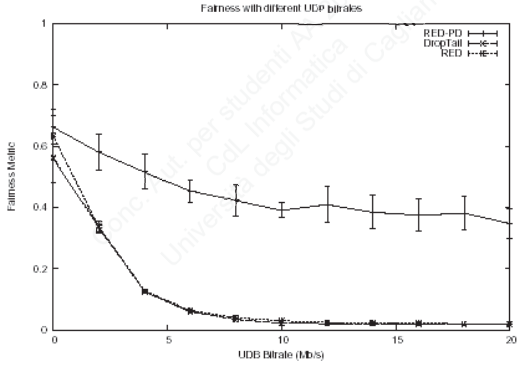

Figura 28. Fairness metric.

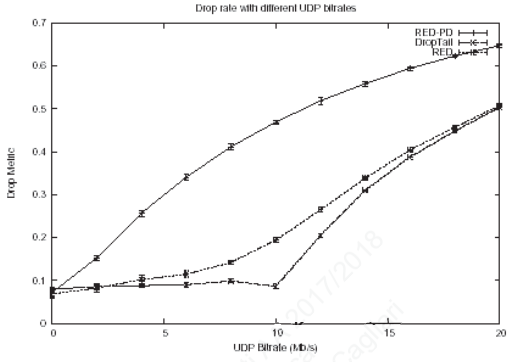

Figura 29. Drop Metric.

Le Figure 28 e 29 mostrano che il più recente miglioramento di RED (RED-PD) raggiunge i suoi obbiettivi prescelti: RED-PD fornisce un bilanciamento più corretto, con più aggressività ottenuta cancellando i pacchetti UDP per primi. Sebbene RED sia più aggressivo del normale Drop-Tail, i grafici mostrano un bilanciamento non dissimile.

bianca

CAPITOLO 5.

QUALITÀ DEL SERVIZIO (QOS).

5.1. INTRODUZIONE.

Nei capitoli precedenti, si è trattato principalmente l'argomento del controllo della congestione, per migliorare le prestazioni di una sottorete.
L'avvento di applicazioni *real time*, e lo sviluppo di applicazioni multimediali, impone misure più restrittive per garantire prestazioni adeguate alle reti di comunicazioni, ovvero si ritiene necessario assicurare una ***Qualità del Servizio*** per applicazione, o complessiva.

5.2. QUALITY OF SERVICE (QOS, QUALITÀ DEL SERVIZIO).

Un insieme di pacchetti diretti da una sorgente ad una destinazione viene chiamato ***flusso*** di una o più sessioni insistenti su quella direttrice.
In una rete connection oriented tutti i pacchetti di un flusso seguono lo stesso percorso. In una rete connectionless i pacchetti possono seguire percorsi differenti. A ciascun flusso è possibile associare una determinata Qualità del Servizio, classificabile con i seguenti parametri:

- *affidabilità,* nessun bit può essere trasmesso in modo scorretto;
- *ritardo*, valutabile e controllabile;

- *jitter*, variazione del tempo di arrivo di ciascun pacchetto (indica pacchetti in arrivo ad intervalli irregolari);
- *banda*, livelli di garanzia, BMG: Banda Minima Garantita.

Applicativi quali la posta elettronica o il telnet necessitano di alta affidabilità. Al contrario, applicazioni *real time*, quali la videoconferenza o la telefonia, necessitano di bassi ritardi, e possibilmente bassi jitter.
In reti ATM, i flussi vengono classificati in categorie, con differenti QoS.
E' possibile utilizzare tali categorie anche per reti differenti da quelle ATM, raggruppandole nelle seguenti quattro:

- *Reti a Velocità Costante* (quali, ad esempio, quelle telefoniche). Fornisce banda uniforme e ritardo uniforme;
- *Reti a Velocità Variabile in tempo reale* (quali la videoconferenza);
- *Reti a Velocità Variabile non in tempo reale* (quali video on-demand);
- *Velocità Disponibile* (per esempio il file transfer), per applicazioni non sensibili al ritardo o allo jitter.

5.3. TECNICHE PER BUONA QUALITÀ DEL SERVIZIO.

5.3.1. BUFFERIZZAZIONE.

Tecnica che consiste nella memorizzazione del flusso, prima della consegna, in un buffer. In tal modo non viene influenzata la banda e l'affidabilità, ma si possono verificare problemi di ritardo, di jitter e di creazione di code anomale.

5.3.2. TRAFFIC SHAPING.

L'idea di questa tecnica è quella di forzare la trasmissione dei pacchetti a un ritmo piuttosto regolare, al fine di limitare il rischio di congestioni.
Gli algoritmi più comuni per l'implementazione del traffic shaping sono:

- leaky bucket (secchio bucato);
- token bucket (secchio di gettoni);
- flow specification (specifica o parametrizzazione del flusso).

ALGORITMO DEL SECCHIO BUCATO (LEAKY BUCKET).

Questo algoritmo si fonda sull'analogia al caso di un secchio riempito d'acqua attraverso un rubinetto con la possibilità di variare la portata istantaneamente, provvisto di un foro sul fondo, che cede l'acqua con portata costante. Se viene immessa troppa acqua, essa fuoriesce dal bordo superiore del secchio e si perde.
Nell'host viene realizzato un Leaky Bucket (Figura 30) autorizzato a riversare sulla rete pacchetti con un fissato *data rate* e che mantiene, nei suoi buffer, quelli accodati per la trasmissione. Se l'host genera più pacchetti di quelli che possono essere contenuti nei buffer, questi vengono perduti.

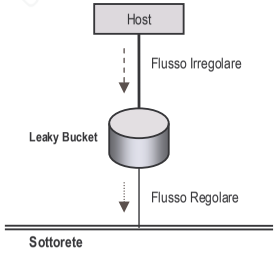

Figura 30. Algoritmo del Secchio Bucato.

ALGORITMO DEL SECCHIO DI GETTONI (TOKEN BUCKET).

E' una tecnica che consente un certo grado di irregolarità nel flusso entrante nella sottorete (Figura 31). Il principio si fonda sull'accumulazione di un credito trasmissivo basato su gettoni (***token***) in fase di non trasmissione, creati a cadenze prefissate, fino ad un massimo consentito (riempimento del secchio di token).
In fase di trasmissione viene sfruttato tale credito fino all'esaurimento, alla massima velocità consentita dalla linea. I pacchetti in eccesso devono aspettare la creazione di nuovi token.
Attraverso tale tecnica è possibile trasmettere dei *burst* di pacchetti, fermo restando che mediamente non si riesce a trasmettere ad una velocità più elevata di quella di generazione dei token.
Inoltre i pacchetti non vengono mai scartati. Se necessario, viene avvertito il livello superiore, produttore dei dati, al fine di interrompere l'invio delle informazioni.

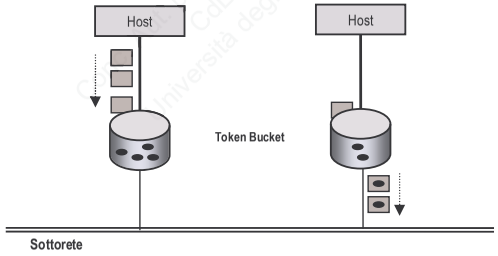

Figura 31. Algoritmo del Secchio di Gettoni.

FLOW SPECIFICATION.

Il traffic shaping è molto efficace se sorgente, sottorete e destinazione si accordano in merito. Un modo di ottenere tale accordo consiste nello specificare:

- le caratteristiche del traffico che si vuole inviare (ad esempio, data rate, grado di *burstiness*);
- la qualità del servizio (ad esempio, ritardo massimo, frazione di pacchetti che si può perdere).

Tale accordo viene identificato con il nome di ***flow specification*** e consiste in una struttura dati che descrive le grandezze in questione.
Sorgente, sottorete e destinatario si accordano di conseguenza per la trasmissione. Questo accordo viene preso prima della trasmissione, e può essere stipulato sia in sottoreti connesse sia in sottoreti non connesse.
In generale nelle reti connesse è più facile il controllo della congestione, perché le risorse per una connessione sono allocate in anticipo. Per evitare la congestione è possibile negare l'attivazione di nuovi circuiti virtuali ove non vi siano sufficienti risorse per gestirli. Questa tecnica va sotto il nome di ***admission control***.

5.3.3. ROUTING ADATTATIVO.

Molti degli algoritmi di routing tentano l'instradamento di tutto il traffico, per ciascuna destinazione, attraverso il percorso migliore. I router non hanno, in generale, una visione completa dei path di traffico di tutta la rete. E' nato così un approccio diverso per garantire una qualità del servizio elevata. Tale metodologia si basa nella suddivisione in percorsi differenti del traffico diretto in ciascuna direzione, utilizzando le informazioni disponibili localmente.

Il metodo più semplice consiste nella suddivisione del traffico in parti uguali nei percorsi alternativi, o proporzionali alla banda disponibile per ciascun percorso. I tal modo, per ogni collegamento diretto, ci si potrà attendere più frazioni di banda uguali ed in grado di essere gestite alla stregua di una multiplazione statistica.

5.3.4. SOVRADIMENSIONAMENTO.

Tecnica che consiste nell'aumentare le capacità di calcolo e le dimensioni del buffer dei router e l'ampiezza di banda per singole tratte della sottorete.
Risulta evidentemente costosa, e, spesso inapplicabile. Tuttavia può rappresentare una valida soluzione se circoscritta nel tempo e per percorsi definiti.

5.4. SERVIZI INTEGRATI.

La gestione di flussi di dati multimediali o *real time*, richiede alla sottorete comportamenti, ai fini della QoS, non assimilabili ai precedenti, soprattutto perché si tratta di traffico generato da applicazioni broadcast, multicast o unicast. Sono nati pertanto specifici algoritmi detti ***algoritmi basati sui flussi*** o ***servizi integrati***. Tali algoritmi sono rivolti principalmente ad applicazioni unicast (traffico molti→ uno, quale ad esempio il download di un videoclip da un sito), o multicast (traffico molti → molti, quali ad esempio trasmissioni video digitali su IP di programmi televisivi).
Il routing broadcast e multicast è già stato trattato nel capitolo 2.
Verrà trattata ora la qualità del servizio in ambiente multicast, tralasciando l'ambiente unicast, in quanto caso particolare del multicast.

5.4.1. PROTOCOLLO RSVP (RESOURCE RESERVATION PROTOCOL).

Molte applicazioni multicast permettono la modifica dinamica dei gruppi di appartenenza e degli utenti di tali gruppi. Pertanto risulta inefficiente un algoritmo in grado di riservare la banda in anticipo per ciascun gruppo, in quanto dovrebbe tenere in considerazione tutti gli ingressi e le uscite di ciascun utente di ciascun gruppo. Per la gestione dei servizi integrati si utilizza pertanto un protocollo chiamato ***RSVP*** (Resource reSerVation Protocol), in grado di occuparsi principalmente della gestione delle prenotazioni, ed ottimizzando l'uso della banda eliminando contemporaneamente le congestioni. Il protocollo utilizza il routing multicast basato su strutture spanning tree. (in Figura 32 è rappresentato lo spanning tree di una rete).

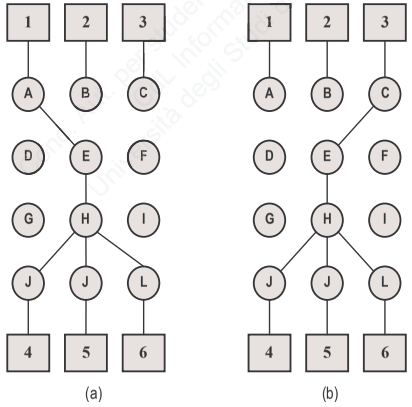

Figura 32. Spanning Tree.

(a). spanning tree multicast host 1.(b). spanning tree multicast host 2.

Ciascun gruppo riceve un indirizzo collettivo. La trasmissione verso un gruppo avviene tramite l'inserimento di tale indirizzo di gruppo nei pacchetti. In tale modo viene costruito lo spanning tree che copre gli elementi del gruppo. Quando l'host 4 richiede un canale all'host 1 viene riservata la banda per la sorgente 1 (Figura 33, disegno a).

Per l'eliminazione delle congestioni tutti i ricevitori inviano un messaggio di prenotazione nella struttura, propagato con l'algoritmo di riverse path forwarding. Il router prende nota della prenotazione ad ogni salto, riservando la banda necessaria. Quando il messaggio torna alla sorgente la banda è stata prenotata lungo il percorso. Successivamente, l'host 4 richiede un secondo canale all'host 2 (Figura 33, disegno b).

Come precedentemente descritto, viene riservato un secondo canale di comunicazione, indipendente da quello precedentemente creato, avente banda prenotata tenendo conto della banda disponibile in quel momento.

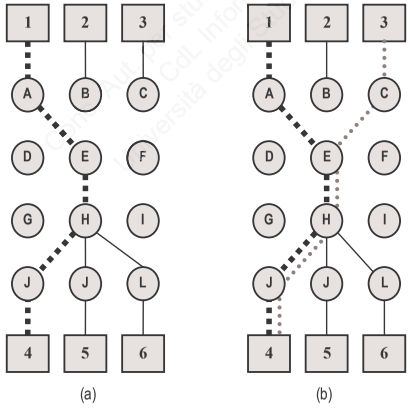

Figura 33. RSVP.

In fase di prenotazione il ricevitore può specificare sorgenti multiple, che possono anche variare lungo la durata della prenotazione stessa, permettendo pertanto un disaccoppiamento tra banda riservata e banda scelta da parte del sorgente.

5.4.2. MPLS (MULTIPROTOCOL LABEL SWITCHING).

Esistono altre metodologie di inoltro dei pacchetti, utili per la gestione dei servizi integrati e differenziati. Una efficiente metodologia consiste nell'aggiungere al pacchetto un'etichetta, al fine di permettere al router l'instradamento del traffico in base ad etichette e non a indirizzi di destinazione, e permettendo in tal modo una gestione più efficace delle risorse.

Tale tecnica prende differenti nomi, incluso ***label switching*** (commutazione di etichetta), o ***tag switching***.

Il nome assegnato dall' IETF, e così standardizzato, è ***MPLS*** (***MultiProtocol Label Switching***).

La posizione dell'inserimento dell'etichetta MPLS sul pacchetto è stata individuata di fronte all'intestazione del pacchetto del livello di rete; nella suite di Internet:IP (Figura 34).

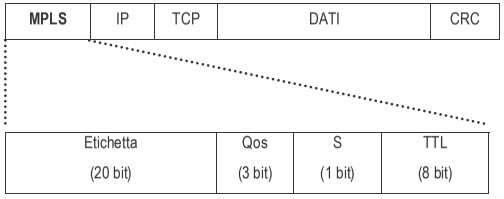

Figura 34. Struttura di un pacchetto TCP/IP con protocollo MPLS.

La sequenza dei campi contenuti nel pacchetto MPLS, fa si che, a rigore nella pila gerarchica ISO-OSI, la sua collocazione debba intendersi tra i livelli di collegamento e di rete; tali atipicità e flessibilità sono forse le ragioni del suo impiego sempre più ampio.

L'intestazione MPLS ha quattro campi:

- *Etichetta* (label): contiene l'indice che serve ad individuare la linea di trasmissione;
- *QoS*: indica la classe di servizio del pacchetto;
- *S*: bit di accatastamento di più etichette; permette di conoscere l'esistenza di etichette incapsulate in altre etichette; il suo valore posto a 1 sta a significare l'appartenenza ad un circuito virtuale, o circuito virtuale concatenato, che il protocollo MPLS rende disponibile anche tra sottoreti diverse tra loro;
- *TTL*, *Time To Live*: viene decrementato ad ogni passaggio su un router; se il suo valore è nullo, il pacchetto viene scartato.

Le intestazioni MPLS sono indipendenti dai livelli successivi e precedenti (Rete e Collegamento). Pertanto è possibile costruire switch MPLS in grado di inoltrare pacchetti di diversi protocolli (MultiProtocol), ad esempio appartenenti alla suite TCP/IP o ATM.

Quando un pacchetto IP, ovvero una cella ATM, con informazioni MPLS, raggiunge un router compatibile, la sua etichetta viene utilizzata come indice per una tabella atta a determinare la linea di trasmissione, e la nuova etichetta da utilizzare.

Due router possono inviare sulla stessa linea di uscita dei pacchetti non in rapporto tra loro, ma con stessa etichetta, creando pertanto dei *circuiti virtuali*.

E' possibile associare un gruppo di etichette a ciascun flusso per la sottorete, o associare un'etichetta collettiva a più flussi. In questo ultimo caso, tali flussi sono chiamati ***Forwarding Equivalence Class*** (***FEC***).

Una differenza rispetto ai circuiti virtuali tradizionali riguarda la possibilità da parte del MPLS di raggruppare più percorsi distinti con punti finali distinti, in quanto i pacchetti contengono oltre all'etichetta, l'indirizzo di destinazione finale. E' perciò possibile rimuovere l'intestazione dell'etichetta alla fine del percorso etichettato, ed inoltrare le informazioni all'indirizzo di destinazione usando le tecniche precedentemente descritte.

MPLS inoltre è in grado di operare su più livelli contemporaneamente, e di generare tunnel ricorsivi attraverso il controllo del bit *S*.

5.6. SERVICE LEVEL AGREEMENT.

Sempre più spesso la garanzia di un apprezzabile Qualità di Servizio è frutto non esclusivo dell'impiego di un solo protocollo ad essa deputato.

Come già trattato in precedenza, nelle metodiche proprie del Traffic Shaping, la QoS è significativamente dipendente da un impiego concordato di risorse non esclusivamente riconducibili ai protocolli attivi sulla sottorete di comunicazione.

Poter commisurare il carico in ingresso in un router al throughput ottimale dello stesso, risulta essenziale concordare il complesso delle prestazioni desiderate dall'Host, o dagli Host, in ingresso sulla sottorete con quelle di fatto erogabili dalla sottorete stessa.

A tal fine, da alcuni si è cercato di regolamentare tale rapporto definendo delle soglie di garanzia minima, o massima, concordata e misurabile su specifici parametri.

La definizione dei livelli di servizio in tal modo introdotti prende nome di **Service Level Agreement (SLA)**.

Le SLA hanno la duplice funzione, verso l'Host e verso la sottorete, di raggruppare al loro interno un insieme di servizi, spesso implementati nei protocolli di livello rete, che rendono misurabili prestazioni specifiche sia dell'Host che della sottorete.

Le SLA hanno una flessibilità di parametrizzazione che ne consente l'impiego con architetture e tecnologie di rete differenti.

Le prestazioni che le SLA standard prevedono vengono assicurate principalmente sulla sottorete, considerando di norma inifluente il contributo del singolo Host.

I livelli di servizio tipicamente considerano:

- *throughput per port* (bit/sec, frame/sec, packets/sec): garanzia sul livello minimo di produttività all'inoltro assicurato per porta del router di sottorete;
- *data delivery ratio*, DDR (bit/sec): garanzia sul livello minimo assicurato di consegna dei dati;
- *constant bit rate*, CBR (bit/sec): garanzia sulla mantenimento di una velocità minima concordata di consegna dei dati per applicazioni partcolari, tipico dell'ATM;
- *back-up di linea* (a caldo, dinamico) (a freddo, statico): garanzia della continuità trasmissiva, in caso di caduta del collegamento principale, con linea di back-up già predisposta (a caldo) o predisposta al momento (a freddo), e, in quest' ultimo caso, i tempi massimi e le risorse stimate per tale operazione;
- *back-up router* (a caldo, dinamico) (a freddo, statico): garanzia della continuità trasmissiva, in caso di malfunzionamento grave o non funzionamento del router di interfacciamento, con router di back-up già predisposto (a caldo) o predisposto al momento (a freddo), e, in quest' ultimo caso, i tempi massimi e le risorse stimate per tale operazione;

- *uptime* (percentuale/mese-anno): garanzia di una percentuale minima stimata di mantenimento del servizio in essere per un tempo di riferimento;
- *report service unit* (RSU): garanzia di poter usufruire di strumenti di controllo delle prestazioni da parte dell'Host, tipicamente non interattivi e di sola visualizzazione;
- *management service unit* (MSU): garanzia di poter usufruire di strumenti di configurazione delle prestazioni da parte dell'Host, tipicamente gestione della banda a propria disposizione, e generazione o modifica di circuiti virtuali permanenti.

La complessità derivante dalla verifica e/o dalla misurabilità dei livelli di servizio delle SLA, hanno portato alla definizione di più semplici e strutturati parametri di indagine denominati **KPI (Key Performance Indicator)** che consentono di raggruppare in un unico *reporting set* l'insieme di KPI necessari alla descrizione del Service Level Agreement desiderato.

BIBLIOGRAFIA.

- R. Adinolfi, "Reti di Computer", McGraw-Hill,1999
- M. Baldi, P. Nicoletti, "Internetworking", seconda edizione , 2004
- D.P. Bertsekas, R. Gallager, "Data networks", seconda edizione, Prentice Hall, 1992
- V. Cerf, R. Kahn, "A protocol for Packet Network Interconnection", IEEE, transation of Communications, Com-22(5), Maggio 1974
- D. Clark, W. Fang, "Esplicit Allocation of Best Effort Packet Delivery Service, IEEE/ACM Transation on Networking, 6(4), Agosto 1998
- D.E. Comer, "Computer Networks and Internet", seconda edizione, Prentice Hall, 1999
- D.E. Comer, "Internetworking with TCP/IP: Principles, Protocols and Architectures", quarta edizione, Prentice Hall, 2000
- D.E. Comer, "Computer Networks and Internets with Internet Applications", terza edizione, Pearson Education, Prentice Hall, 2001
- S.E. Deering, D.R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", ACM Transations on Computer Systems, 8(2), Maggio 1990
- S.E. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.G. Liu, L. Wei, "An Architecture for Wide Area Multicasting Routing", Proceedings of ACM SIGCOMM, 1994
- G. Falk, "The Structure and Funtion of Network protocols", in W. Chou, Computer Communications, vol. I, Principles, Prentice Hall, Upper Saddle River, New Jersey, 1983
- H. Frank, W. Chou, "Routing in Computer Networks", Networks 1(1), 1971

- F. Halsall, "Reti di calcolatori e sistemi aperti", quarta edizione., Addison Wesley-Pearson Education, 1998
- C. Huitema, "Routing in the Internet", Prentice Hall, 1995
- V. Jacobson, "Congestion Avoidance and Control", Proceedings of ACM SIGCOMM, 1988
- R. Jain, "Myths About Congestion Management in High-Speed Networks", Internetworking: Research and Experience, 3(3), Maggio 1992
- J.F. Kurose, K.W. Ross, "Internet e Reti di Calcolatori", McGraw Hill, 2003
- J.F. Kurose, K.W. Ross, "Computer Networking: A Top-Down Approach Featuring the Internet", Addison Wesley Longman, 2001
- Y.Lin et al., "QoS Routing Granularity in MPLS Network", IEEE Comm. Magazine, giugno 2002
- Y. Liu et al. , "Scalable Fluid Models and Simulations for Large-Scale IP Networks", ACM Transactions on Modeling and Computer Simulation, Vol. 14, No. 3, Luglio 2004.
- T. Narten, "Internet Routing", Proceedings of ACM SIGCOMM, 1989
- S. Nelakuditi et Z. Zhang, "A Localized Adaptive Proportioning Approach to QoS Routing", IEEE Comm. Magazine, giugno 2002
- R. Puzmanova, "Routing and Switching: time of convergence?", Addison-Wesley, 2002
- R.W. Smith, "Broadband Internet Connections", Addison Wesley, 2002
- W. Stallings, "Data and Computer Communications", Mac Millan Publishing, 1995
- W. Stallings, "High Speed Networks: TCP/IP and ATM Design Principles", Prentice Hall, Upper Saddle River, New Jersey, 1998
- A.S. Tanenbaum, "Reti di Calcolatori", quarta edizione, Pearson Prentice Hall, 2003
- Z. Wang, "Internet QoS", Morgan Kaufmann, 2001
- D. Wetteroth, "OSI Reference Model for telecommunications", McGraw Hill, 2001
- L. Zhang, "RSVP: A new reserouce reservation protocol", IEEE Network Magazine, ottobre 1993