RETI di TLC

M. Ajmone Marsan
F. Neri

appunti delle lezioni

con contributi di:

Andrea Bianco
Claudio Casetti
Emilio Leonardi
Renato Locigno
Marco Mellia
Michela Meo
Antonio Nucci

Indice

1

Introduzione alle reti di telecomunicazioni

1.1 Generalità

Una rete di telecomunicazioni è un sistema che fornisce *servizi* relativi al *trasferimento di informazioni* ad una popolazione di *utenti* distribuiti geograficamente.

Le reti di telecomunicazioni sono vicine alla nostra esperienza quotidiana di uomini moderni: basti pensare alla rete telefonica, alla rete postale, alle reti per diffusione radio e TV, alle reti telematiche.

Alcune di queste reti sono di nuova concezione e quindi utilizzano tecnologie avanzate, tipicamente del settore elettronico (e in qualche caso anche della fotonica), mentre altre, come la rete postale, sono state in funzione per quasi due secoli e si basano su strumenti molto più tradizionali, quali i mezzi di trasporto.

Sappiamo inoltre che in tempi remoti sono esistite reti di telecomunicazioni basate su tecnologie diverse, come torri di avvistamento e segnali luminosi o bandiere (i castelli della Valle d'Aosta, la Grande Muraglia Cinese), segnali di fumo (caratteristici degli indiani americani), o segnali acustici (i tam-tam della giungla). Inoltre, verso la fine del secolo scorso erano state attivate reti telegrafiche basate su segnalazioni ottiche, utilizzando tralicci su cui venivano montati pannelli mobili azionabili dal basso e visibili da lontano.

È evidente una significativa differenza tra le reti citate ad esempio: le reti per diffusione radio e TV, i segnali di fumo ed i tam-tam costituiscono reti a *diffusione* e *unidirezionali*: infatti l'informazione viene distribuita da una sorgente a chiunque disponga di un apparato ricevitore, quindi a ogni utente della rete, indipendentemente dalla sua identità (diffusione). Non è inoltre possibile per la gran maggioranza degli utenti, che dispongono solo di un apparato ricevente, inviare informazioni ad altri (unidirezionalità).

Le reti telematiche, la rete telefonica, postale, sono invece reti a *selezione* e *bidirezionali*, infatti esse sono caratterizzate dalla possibilità per la sorgente dell'informazione di scegliere a quali interlocutori questa deve essere trasferita (selezione). Inoltre, tipicamente tutti gli utenti sono attrezzati sia per trasmettere sia per ricevere (bidirezionalità).

In queste dispense ci interesseremo esclusivamente di reti realizzate con tecnologie elettroniche e prevalentemente di reti a selezione e bidirezionali (anche se vedremo alcune eccezioni). Ci occuperemo infatti di reti telefoniche e (soprattutto) di reti telematiche, limitandoci al caso di trasmissione di informazione numerica (o numerizzata).

1.2 Cenni storici

Le origini delle reti di telecomunicazioni con tecnologia elettronica si possono far risalire fino alle invenzioni del telegrafo e del telefono. La datazione dell'invenzione del telegrafo non è semplice, ma si può prendere come riferimento l'anno della proposta del codice Morse (1837). Più facile è invece la datazione dell'invenzione del telefono (nonostante la diatriba tra Bell e Meucci), che si può far risalire al brevetto di Bell, depositato nel 1876.

Le prime reti telefoniche furono realizzate collegando gli apparecchi di utente a centrali in cui il collegamento tra utente chiamante ed utente chiamato (la *commutazione*) veniva realizzato manualmente da operatori, su richiesta dell'utente chiamante. Questi, dopo avere avvisato l'operatore dell'intenzione di inoltrare una richiesta (mediante un segnale acustico o luminoso), specificava a voce con chi desiderava essere collegato (fornendo una informazione detta di *segnalazione*). La topologia delle reti in questo caso era di tipo monocentrico (anche detto *stellare*), con una unica centrale a cui tutti gli utenti erano collegati.

La diffusione del servizio telefonico portò da una parte a topologie di rete più complesse (policentriche) e dall'altra all'invenzione di apparati elettromeccanici per l'automazione delle funzioni di commutazione. È del 1891 il brevetto del selettore Strowger (dal nome del suo inventore, un impresario di pompe funebri!), che permetteva con movimenti di sollevamento e rotazione la selezione di uno tra molti contatti disposti su di una superficie cilindrica. Nel 1984 fu messa in servizio la prima centrale urbana che incorporava queste meraviglie della tecnologia elettromeccanica e nel 1923 fu realizzato in Baviera il primo servizio interurbano automatico con teleselezione di utente (con apparati Siemens).

In queste reti la segnalazione avveniva sui canali di comunicazione che venivano via via utilizzati per il collegamento tra utente chiamante ed utente chiamato (segnalazione associata).

La tecnologia elettromeccanica dominò il panorama delle reti telefoniche per vari decenni, sostituendo progressivamente i selettori Strowger con reti di connessione "crossbar" e con relè, ottenendo significativi vantaggi in termini di miniaturizzazione.

Fu solo negli anni '60 che le tecnologie elettroniche fecero il loro ingresso nel settore della commutazione, inizialmente come sistemi di controllo di apparati elettromeccanici. È del 1964 l'installazione a Succasunna, negli USA, della prima centrale controllata da un elaboratore elettronico.

Con l'intervento degli elaboratori fu possibile separare la segnalazione dal flusso dell'informazione, guadagnando in efficienza (segnalazione quasi associata o dissociata).

Circa 10 anni più tardi gli elaboratori elettronici diventavano lo strumento non solo per il controllo, ma anche per l'attuazione delle funzioni di commutazione (Chicago, 1975, prima centrale pubblica interamente numerica).

In parallelo all'introduzione di tecnologie numeriche nel controllo delle centrali e nell'attuazione delle funzioni di segnalazione e commutazione, si è assistito all'introduzione di tecnologie numeriche nella trasmissione del segnale vocale, che viene codificato con tecniche varie (solitamente PCM – Pulse Coded Modulation) in una sequenza di cifre binarie e multiplato sui mezzi trasmissivi con schemi a divisione di tempo (TDM – Time Division Multiplexing).

Ancora oggi, in molti paesi del mondo, la conversione verso la tecnologia interamente numerica deve essere completata. Intanto, con la disponbilità di grandi capacità di elaborazione in rete, sono stati sviluppati servizi addizionali (numero verde, chiamata in attesa, trasferimento di chiamata, conferenza, eccetera) e si stanno sviluppando le prime reti intelligenti.

La reti interamente digitali permettono una tecnica più sofisticata ed intelligente di segnalazione, detta segnalazione *a canale comune*, in cui si usa una rete di telecomunicazioni numerica per la segnalazione sovrapposta alla rete per il trasporto dei segnali vocali.

Il passo successivo che è in corso in questi anni consiste nell'integrazione dei servizi voce e dati in una sola rete, denominata ISDN (Integrated Services Digital Network). In alcuni paesi l'ISDN è una realtà già da

tempo, in altri si è appena agli inizi, in altri ancora non si è ancora iniziato. Il primo paese ad introdurre un servizio ISDN commerciale è stata la Francia nel 1987. In Italia il servizio pilota ISDN ha avuto inizio nel 1991.

Un'altra evoluzione in corso nel settore delle reti telefoniche riguarda la telefonia mobile cellulare, che dopo i primi sistemi analogici sta ora migrando verso quelli numerici (è del 1988 lo standard GSM).

Gli ulteriori sviluppi che sono prevedibili a medio termine riguardano da un lato l'integrazione di servizi ad alta velocità in una rete denominata B-ISDN (Broadband ISDN) e dall'altro l'evoluzione verso terminali portatili universali (UPC – Universal Personal Communications). Per il lungo termine si stanno investigando le possibilità di impiego delle tecnologie fotoniche nel settore della commutazione (oltre che in quello della trasmissione, che è ormai una realtà).

A fronte di questo tumultuoso sviluppo delle reti telefoniche, le reti pubbliche per dati hanno avuto una significativa evoluzione propria solo molto più di recente, sull'onda della diffusione sempre più capillare degli elaboratori elettronici. Le prime reti per dati in area geografica sono state reti private, che sfruttavano i portanti delle reti pubbliche telefoniche instaurando circuiti numerici mediante modem opportuni. I primi esperimenti di reti di calcolatori su scala geografica sono probabilmente riconducibili al progetto ARPANET, iniziato nel 1969. Le reti pubbliche di calcolatori si sono sviluppate in ambito nazionale ed internazionale, dando origine agli standard X.25 (del 1976 la prima versione, del 1980 la seconda), OSI (1980) e Frame Relay.

In parallelo alle reti pubbliche per dati, sono nate e si sono molto diffuse in ambito privato le reti di calcolatori in area locale (LAN – Local Area Network: Ethernet è del 1972, il token ring IBM del 1982, FDDI del 1985) ed in area metropolitana (MAN – Metropolitan Area Network: QPSX è del 1986 e lo standard IEEE 802.6 – DQDB – del 1991).

Solo di recente la rete Internet, emanazione degli esperimenti ARPANET, ha trovato larghissima diffusione sia in ambito privato sia in ambito pubblico, favorita dall'introduzione di applicativi di distribuzione dell'informazione (principalmente il WWW – World Wide Web).

1.3 I servizi e le loro caratteristiche di traffico

Alcuni dei servizi che le reti di telecomunicazioni offrono ai loro utenti possono essere elencati come segue:

- telefonia (privata e pubblica, fissa e mobile)
- facsimile
- videoconferenza
- teledrin
- trasferimento di segnali video (video lento, segnali TV, video di alta qualità, compressi o non compressi)
- trasmissione di immagini
- trasferimento di files
- posta elettronica
- accesso remoto a elaboratori
- accesso a basi dati

- telesorveglianza
- telecontrollo
- telemedicina
- teledidattica
- home banking
- moneta elettronica
- telemarketing
- calcolo distribuito
- lavoro collaborativo
- telegioco e gioco collaborativo
- ...

I diversi tipi di servizio producono diversi tipi di caratteristiche dei dati che devono essere trasportati dalla rete, o come si dice in gergo, diversi tipi di *traffico*. Inoltre, i diversi servizi pongono requisiti diversi sul funzionamento della rete.

Nella figura 1.1 sono indicati i requisiti di alcuni dei servizi citati, per ciò che riguarda il volume dell'informazione da trasferire ed il ritardo tollerato nel trasferimento. Combinando i due dati si ottengono delle indicazioni sulla velocità di trasmissione equivalente che la rete deve fornire per l'erogazione del servizio.

La caratterizzazione del traffico e dei suoi requisiti è di fondamentale importanza per poter procedere ad un dimensionamento della rete e al progetto delle procedure operative interne alla rete, al fine di soddisfare al meglio le necessità degli utenti.

Una prima caratterizzazione del traffico generato da un servizio riguarda la *periodicità* nella generazione dei dati da trasmettere da parte della sorgente. Distinguiamo così traffico di tipo:

- periodico
- aperiodico od *impulsivo*

Esempi di traffico periodico sono forniti dai segnali numerici risultanti dalla digitalizzazione di segnali vocali o video. Per esempio, nel caso di numerizzazione del segnale vocale con tecnica PCM viene prodotto un byte di informazione ogni 125 μs. Un tipico esempio di traffico impulsivo è prodotto da applicazioni di tipo transazionale in reti telematiche: i messaggi prodotti da un utente che interroga una banca dati sono generati ad intervalli variabili ed hanno dimensioni variabili.

Si noti che considerare periodico il traffico prodotto dalla numerizzazione di un segnale vocale o video equivale ad esaminare solo il periodo in cui il servizio è attivato, trascurando la dinamica temporale relativa all'apertura e chiusura di una conversazione telefonica o di un collegamento video. Ovviamente, se invece si considerano le caratteristiche relative agli istanti di inizio del servizio ed alla sua durata complessiva, il traffico presenta caratteristiche di impulsività. Il livello di dettaglio usato nell'analisi o nel progetto della rete determinano quale sia la caratterizzazione del traffico più adeguata per un determinato tipo di servizio. Nel caso del servizio di telefonia, le costanti di tempo relative alla apertura e chiusura dei collegamenti sono di parecchi ordini di grandezza superiori rispetto a quelle caratteristiche della generazione dell'unità dati elementare. Ne consegue che un'analisi di dettaglio tenderà a considerare il traffico telefonico come periodico, mentre un'analisi più astratta considererà le caratteristiche impulsive delle stesso tipo di traffico.

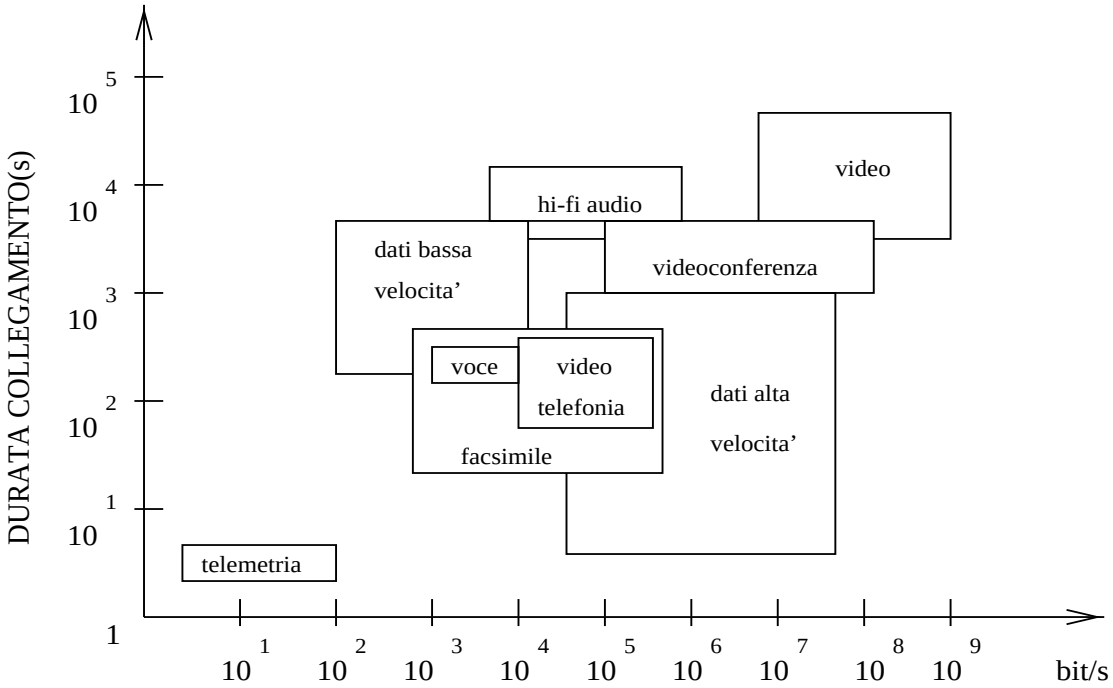


Figura 1.1. Caratteristiche di alcuni servizi di telecomunicazioni

Anche nel caso di traffico impulsivo, considerando un livello di dettaglio opportuno, in certi casi è possibile individuare caratteristiche di periodicità. Si pensi ad esempio al trasferimento di un file in una rete telematica (per esempio la risposta ad una interrogazione di una banca dati). In questo caso le richieste avvengono in modo saltuario, ma una richiesta corrisponde alla generazione di molte unità dati elementari con un periodo che è quello relativo all'apparato da cui viene prelevata l'informazione (ad esempio un disco di un elaboratore elettronico su cui il file risiede).

Una seconda caratterizzazione del traffico riguarda la necessità di mantenere in ricezione le stesse relazioni temporali con cui i dati sono stati prodotti dalla sorgente. Distinguiamo quindi traffico

- isocrono
- anisocrono (o asincrono)

Esempi tipici di traffico isocrono sono forniti dai segnali risultanti dalla numerizzazione di segnali analogici. Infatti, la necessità di ricostruire il segnale analogico per la riproduzione alla destinazione (si pensi ancora una volta al segnale vocale o al segnale video) fa sì che si debba disporre dei campioni con cadenza prefissata. Se un campione non è disponibile al momento in cui deve essere utilizzato per la conversione D/A, si deve interrompere l'uscita.

Nel caso del trasferimento di un file o di un'interrogazione di una banca dati invece non si pongono requisiti così stringenti: il ritardo di una parte dell'informazione non è significativo; ciò che importa è il tempo totale in cui si riesce a completare una transazione.

La terza classificazione del traffico riguarda la necessità di integrità dell'informazione. Mentre è ovvio che si desidera sempre trasferire l'informazione con la migliore qualità possibile, si deve distinguere il caso

in cui l'integrità dell'informazione è indispensabile per il servizio, dal caso in cui la ridondanza intrinseca dell'informazione da trasferire fa sì che si possa tollerare la perdita di una piccola frazione dell'informazione senza degradare significativamente la qualità del servizio.

Ancora una volta l'esempio migliore di traffico che tollera la perdita di una piccola frazione dell'informazione è fornito dai segnali risultanti dalla numerizzazione di segnali analogici. Per esempio, nel caso di voce numerizzata con tecniche PCM, la perdita di alcuni byte porta solo ad un lieve disturbo nella ricezione, senza impedire la comprensione del significato del messaggio. Al contrario, nel caso di trasferimento di un file o di un'interrogazione ad una banca dati, la presenza di anche un solo byte errato rende sovente inutilizzabile l'informazione.

La distinzione tra i diversi tipi di traffico è rilevante anche per la definizione degli indici di prestazione adeguati a misurare la qualità del servizio fornito dalla rete agli utenti.

Ad esempio, nel caso di servizi che generano traffico periodico, isocrono, con tolleranza di perdita (per esempio telefonia o trasmissione di video numerico) sono rilevanti indici di prestazione quali la probabilità di blocco (cioè la probabilità di non poter esaudire una richiesta di collegamento), il ritardo massimo e/o la variazione del ritardo introdotto dalla rete e la probabilità di perdita di un byte. Questi ultimi due parametri possono essere valutati congiuntamente conoscendo la probabilità con cui un certo valore di ritardo viene superato.

Per servizi che generano traffico impulsivo, anisocrono, con requisiti di integrità (tutti i servizi di tipo transazionale: interrogazione di banche dati, posta elettronica, accesso remoto ad elaboratori, eccetera) è più importante conoscere parametri quali il ritardo medio e la probabilità di errore residua sui bit consegnati all'utente.

Infine, per servizi quali il trasferimento di un file di grandi dimensioni, l'indice di prestazione più rilevante può essere la quantità di informazione mediamente trasferita nell'unità di tempo, che viene chiamata traffico smaltito (o anche "throughput").

1.4 Funzioni di una rete di telecomunicazioni

Le principali *funzioni* di una rete di telecomunicazioni sono classificabili come segue:

- commutazione,
- trasmissione,
- segnalazione,
- gestione.

La *commutazione* ha il compito di selezionare le opportune risorse di rete per far comunicare i due (o più) utenti coinvolti nello scambio di informazioni.

La *trasmissione* ha il compito di realizzare l'effettivo trasferimento dell'informazione numerica dalla sorgente ai (al) destinatari(o).

La *segnalazione* ha il compito di trasferire dall'utente alla rete e tra i vari organi di commutazione all'interno della rete le informazioni che sono necessarie per la funzione di commutazione.

La *gestione* ha il compito di amministrare le risorse della rete al fine di permetterne l'esercizio e la manutenzione, nonchè di implementare il controllo di qualità e la supervisione del traffico.

In questo testo focalizzeremo la nostra attenzione in modo particolare sulle funzioni di commutazione e segnalazione.

1.5 Commutazione in reti numeriche

Esistono sostanzialmente due diverse tecniche di commutazione

1. commutazione di circuito,
2. commutazione di messaggio e di pacchetto.

Nel seguito si descriveranno in modo molto schematico i principi di funzionamento delle due tecniche, insieme alle motivazioni che hanno portato alla loro utilizzazione in settori diversi.

1.5.1 Commutazione di circuito

Quando un utente A desidera comunicare con un utente B, l'utente A fornisce alla rete l'informazione (detta *informazione di segnalazione*) necessaria per avviare la costruzione del canale (o *circuito*) utilizzato per collegare A e B.

La costruzione del circuito viene effettuata dalle centrali di commutazione che a loro volta si scambiano informazioni relative al circuito da instaurare (anche questa informazione fa parte della segnalazione).

Una volta costruito, il circuito è di uso esclusivo degli interlocutori per tutta la durata della comunicazione e viene rilasciato (si dice anche "abbattuto") solo su indicazione di uno dei due.

Il procedimento si può generalizzare al caso di un collegamento di più di due utenti.

Questa tecnica di commutazione è stata ideata e sviluppata per le reti di tipo telefonico ed in tale ambito è stata lungamente collaudata. Motivi di efficienza sconsigliano però l'utilizzazione di questa tecnica di commutazione per reti dati ad alta velocità. Per giustificare questa affermazione impostiamo, sia pur in modo estremamente semplificato, un calcolo dell'efficienza della commutazione di circuito.

Chiamiamo C il tempo necessario per costruire il circuito, D il tempo impiegato per trasferire i dati e R il tempo necessario per il rilascio del canale. Definiamo l'efficienza della tecnica di commutazione η come

$$\eta = \frac{D}{C + D + R}$$

Facendo riferimento ad una rete telefonica, C può assumere valori che vanno da frazioni di secondo fino a 5 secondi; 5 secondi è il limite massimo trascorso il quale si decide che il collegamento non possa essere effettuato e si invia all'utente il segnale di occupato. La fase di rilascio è più veloce ed R può assumere valori che vanno dalle frazioni di secondo al secondo. Quindi si può grosso modo supporre che la parte accessoria alla trasmissione dei dati (cioè $C + R$) richieda qualche secondo. Per i nostri calcoli prendiamo $C + R$ uguale a 2 secondi (un valore intermedio).

Supponiamo di volere trasmettere un file di dati della dimensione di 1000 byte (quindi un file di piccole dimensioni).

Per ottenere $\eta = 0.98$ si deve avere $D = 98$ secondi, il che si ottiene con una velocità di trasmissione pari a $8000/98 = 81.6$ bit al secondo. Se scegliamo una velocità di trasmissione più ragionevole, dell'ordine di 10 Kbit/s, l'efficienza si riduce a $\eta = 0.286$. Se poi si sceglie una velocità di trasmissione elevata, quale può essere 10 Mbit/s, il valore dell'efficienza diventa un ridicolo $\eta = 0.0004$.

È ovvio che una situazione che permette di raggiungere efficienze accettabili solo in presenza di velocità di trasmissione ridicolmente basse non è sostenibile. Bisogna individuare delle strategie alternative. Due sono le strade che si possono seguire:

1. la prima strada è quella di ridurre i valori di C ed R drasticamente, di alcuni ordini di grandezza; cosa non banale o addirittura impossibile in reti di grandi dimensioni, ma fattibile in ambito locale, per esempio in centraline di commutazione private;

2. la seconda strada è quella di modificare l'approccio al problema della commutazione, rinunciando alla costruzione di un circuito di uso esclusivo dei due interlocutori, passando quindi dalla commutazione di circuito alla commutazione di messaggio o di pacchetto.

1.5.2 Commutazione di messaggio

Quando un utente A deve inviare informazioni ad un utente B, A formatta l'informazione in un unico *messaggio*. Il messaggio viaggia nella rete da una centrale (o nodo) di commutazione all'altra seguendo un itinerario calcolato volta per volta (usando la funzione dell'*instradamento* che è una delle componenti principali della commutazione) finchè non arriva a destinazione. In ogni nodo di commutazione il messaggio viene normalmente ricevuto per intero prima di essere ritrasmesso verso il prossimo nodo. Si parla di funzionamento di tipo *store-and-forward:* i messaggi vengono memorizzati e poi inoltrati verso il nodo successivo.

Un esempio di funzionamento in qualche misura simile si può riconoscere nella rete postale, dove si utilizzano volta per volta diversi mezzi di comunicazione per fare arrivare a destinazione una lettera.

Con questo approccio si elimina la perdita di efficienza tipica della commutazione di circuito perché non si perde più tempo per creare il canale. Inoltre non si alloca una risorsa in modo statico ad una coppia di interlocutori, ma tutti i canali della rete sono utilizzati per la spedizione di messaggi.

Questi vantaggi si pagano con la necessità di calcolare per ogni messaggio il percorso da seguire (instradamento) e con la necessità di accodare i messaggi per la trasmissione qualora un canale sia già occupato con la trasmissione di un altro messaggio.

La perdita di efficienza rispetto al caso ideale è ora dovuta al fatto che il messaggio non è composto solamente dai dati corrispondenti all'informazione da trasferire, ma a questi viene aggiunta una *intestazione* (detta anche "header"), contenente informazione di controllo quale l'indirizzo del destinatario del messaggio, e sovente anche una coda ("tail"), contenente bit di ridondanza, che aumentano la dimensione dei dati da trasmettere. Ciò non avveniva nella commutazione di circuito perchè vi era un collegamento diretto e quindi non c'era il bisogno di dire alla rete chi era il destinatario dell'informazione trasmessa. I bit di ridondanza aggiunti nella coda permettono di offrire servizi di qualità superiore a quella della commutazione di circuito, implementando per esempio un controllo per la protezione dagli errori di trasmissione. L'intestazione di ogni messaggio contiene dell'informazione di segnalazione, che ogni centrale utilizza per svolgere le operazioni di commutazione (cioè per decidere l'instradamento del messaggio).

Il fatto che l'instradamento sia calcolato messaggio per messaggio in dipendenza dalle condizioni istantanee della rete fa sì che due messaggi trasmessi in sequenza possano arrivare in ordine inverso, ed in generale che siano possibili ritardi fortemente variabili. È persino possibile che un messaggio venga perso o addirittura duplicato.

1.5.3 Commutazione di pacchetto

La commutazione di pacchetto è sostanzialmente simile alla commutazione di messaggio, ma i messaggi lunghi sono frammentati in blocchi più piccoli detti *pacchetti*. Ad ogni pacchetto vengono aggiunti una intestazione e possibilmente una coda, contenenti informazioni di controllo.

Diversi sono i motivi che ci inducono a questa soluzione. Un motivo risiede nel fatto che se sul canale di trasmissione si ha una probabilità di errore p e i pacchetti sono lunghi k bit, la probabilità di corretta ricezione sul pacchetto è data da $P = (1-p)^k$, che per k grande tende a diventare piccola, indipendentemente dal valore di p. Per $k \to \infty$, $P \to 0$ in maniera esponenziale. Non conviene allora avere messaggi lunghi, ma conviene frazionarli in pacchetti.

Un secondo motivo che ci induce a frazionare il messaggio in pacchetti è dovuto alla possibilità di trasmissione in "pipeline". Ciò significa che non appena è ultimata la trasmissione del primo pacchetto sul

primo canale si può iniziare ad inoltrarlo sul secondo canale, mentre sul primo canale è in transito il secondo pacchetto. Ciò corrisponde a ridurre il minimo tempo necessario al messaggio per attraversare un nodo di commutazione: visto che il funzionamento store-and-forward prevede che il messaggio venga ricevuto per intero prima che il primo bit inizi ad essere trasmesso sul canale successivo, messaggi più corti riducono il ritardo di commutazione.

Un esempio di funzionamento della commutazione di pacchetto confrontato con la commutazione di circuito e la commutazione di messaggio nello stesso scenario è mostrato nelle figure 1.2, 1.3 e 1.4. In tali figure, note come diagrammi spazio-temporali, il tempo evolve dall'alto verso il basso, mentre lo spazio evolve orizzontalmente. Utilizzeremo sovente tali schemi per descrivere il funzionamento dei protocolli.

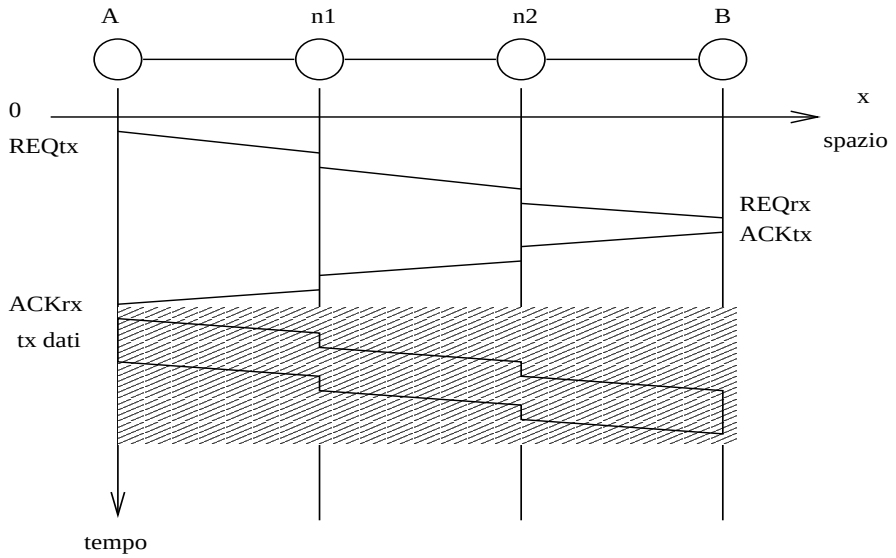


Figura 1.2. Commutazione di circuito

Il vantaggio della commutazione di pacchetto rispetto alla commutazione di messaggio è dovuto al fatto che mentre con la seconda si usano diversi canali in tempi di trasmissione disgiunti, con la prima si ha un parallelismo dovuto al fatto che i pacchetti possono essere inoltrati verso la destinazione prima che il messaggio sia stato trasmesso completamente.

Lo svantaggio della commutazione di pacchetto sta nel fatto che è aumentata l'informazione di controllo da trasmettere a causa dell'aggiunta di intestazioni e code per ogni pacchetto. Per la ricerca della lunghezza ottima dei pacchetti si deve cercare il compromesso più conveniente tra la probabilità di errore, che è minimizzata da pacchetti corti, e la quantità totale di bit trasmessi.

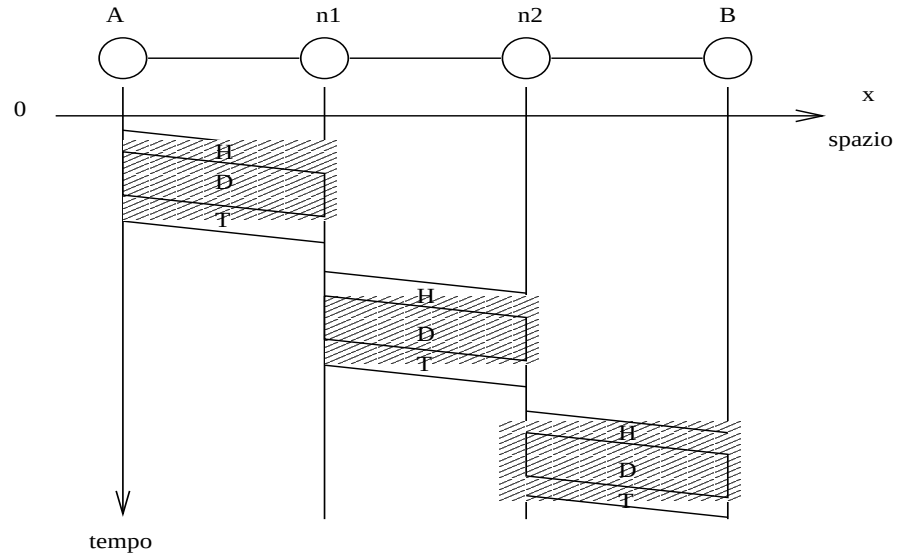


Figura 1.3. Commutazione di messaggio

1.5.4 I circuiti virtuali

Sovente nelle reti di telecomunicazioni si sovrappone alla tecnologia della commutazione di pacchetto appena descritta il meccanismo dei “circuiti virtuali” (VC – anche detti connessioni).

I vantaggi derivanti dall’introduzione dei circuiti virtuali sono legati ad una maggior semplicità nell’instradamento ed al fatto che tutti i pacchetti di uno stesso VC seguono lo stesso percorso nel loro viaggio dalla sorgente alla destinazione.

La semplificazione dell’instradamento deriva dal fatto che ora è necessario calcolare l’instradamento solo al momento dell’instaurazione del VC; tutti i pacchetti che ad esso si riferiscono utilizzeranno poi l’instradamento precalcolato una volta per tutte. Ciò comporta anche delle semplificazioni nell’indirizzamento: infatti l’identificatore del VC corrisponde ad una coppia di indirizzi sorgente/destinazione, quindi è possibile ridurre l’informazione di controllo contenuta in ogni singolo pacchetto.

Il fatto che i pacchetti seguano tutti uno stesso percorso significa che è facile mantenerne la sequenza, cosa che invece diventa impossibile se pacchetti diversi seguono strade diverse per raggiungere la destinazione. Mantenere la sequenza può essere assai più conveniente che ricostruirla al destinatario.

Il fatto che il circuito sia virtuale e non fisico permette di ottenere i vantaggi citati senza incorrere nelle perdite di efficienza dovute alla allocazione di un circuito per uso esclusivo di due utenti. Cionondimeno l’introduzione dei VC comporta la ricomparsa dei ritardi necessari per la costruzione del VC e per il suo rilascio.

Inoltre va sottolineato il fatto che i VC non permettono di ottenere ritardi fissi tra sorgente e destinazione, risultando così inadatti al trasferimento di traffico isocrono.

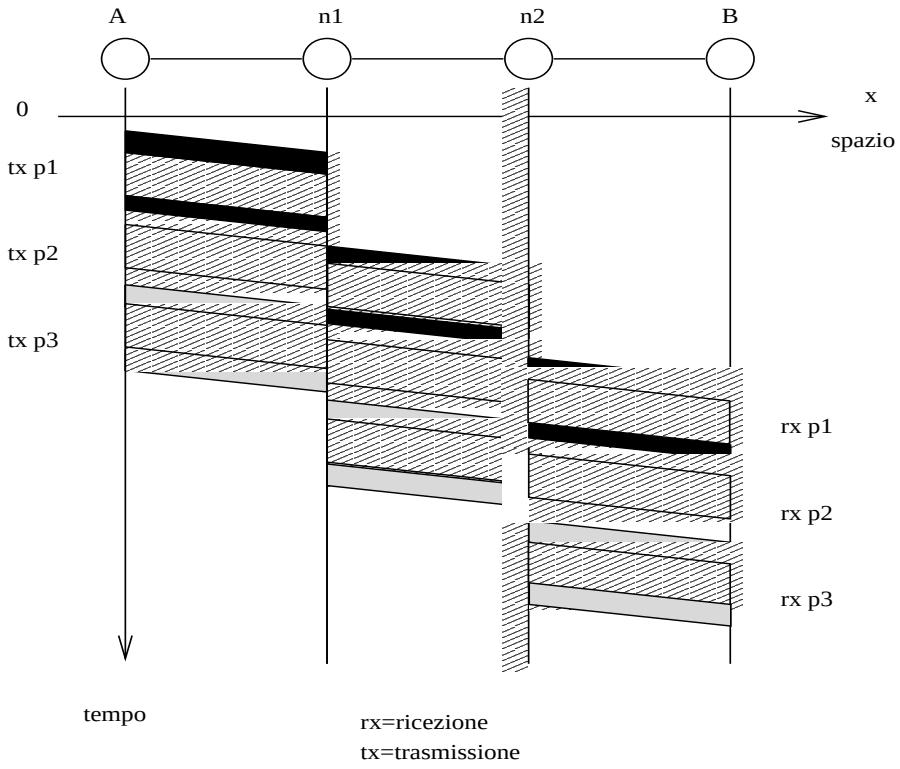


Figura 1.4. Commutazione di pacchetto

1.5.5 Considerazioni sulle tecniche di commutazione

Introduciamo in questo paragrafo alcune considerazioni generali nella valutazione delle tecniche di commutazione descritte nei paragrafi precedenti.

La commutazione di circuito implica completa trasparenza, quindi anche stessa velocità di trasmissione tra canali entranti e canali uscenti in un nodo di commutazione. Visto che sovente la banda disponibile sui collegamenti della rete non è sempre la stessa, su ogni canale la banda viene partizionata, utilizzando opportune tecniche di multiplazione, in un certo numero di canali trasmissivi (che hanno di solito la stessa velocità in tutta la rete). Una comunicazione occupa una concatenazione di canali tra la sorgente e la destinazione dell'informazione. Nel caso di commutazione di pacchetto, invece, la banda sui collegamenti resta di solito indivisa e viene allocata dinamicamente ai vari pacchetti che, in divisione di tempo statistica, attraversano il canale. Abbiamo quindi sovente canali di ingresso e di uscita di velocità differenti, il che implica una commutazione in modalità store-and-forward. In generale mantenere la banda indivisa garantisce migliori prestazioni in termini di ritardo, come verrà discusso nel capitolo sui sistemi a coda.

Generalizzando la nozione di commutazione esposta nei paragrafi precendenti, possiamo affermare che la commutazione è il processo di allocazione delle risorse (banda, memoria, eccetera) necessarie ad effettuare una comunicazione. Nel caso di commutazione di circuito, abbiamo una allocazione *totale* e *preventiva* delle risorse necessarie, prima di iniziare il vero e proprio trasferimento di informazione. Nel caso di commuta-

zione di pacchetto senza circuiti virtuali, non abbiamo alcuna allocazione preventiva, e le risorse vengono allocate dinamicamente tratta per tratta lungo il percorso tra la sorgente e la destinazione: abbiamo un'allocazione parziale e progressiva di risorse. Se le risorse non sono disponibili, nel caso di commutazione di circuito la richiesta di creazione del circuito viene rifiutata, mentre, nel caso di commutazione di pacchetto, i pacchetti vengono temporaneamente memorizzati nei nodi di commutazione in attesa di avere disponibilità di risorse. Se tale memorizzazione non è possibile per assenza di memoria (fenomeni di congestione), i pacchetti vengono di solito scartati. Nel caso di commutazione di pacchetto con circuiti virtuali, si ha una fase iniziale di richiesta del circuito virtuale, che è molto simile alla commutazione di circuito. In tale fase è possibile (ma non necessario) effettuare una totale (o parziale) allocazione delle risorse richieste sul cammino tra la sorgente e la destinazione; è inoltre possibile rifiutare la richiesta di circuito virtuale. In un certo senso, i circuiti virtuali forniscono una tecnica di commutazione intermedia tra la commutazione di circuito e di pacchetto, avvicinandosi all'una o all'altra a seconda di quante risorse sono preallocate nella fase di creazione del circuito virtuale.

La commutazione di circuito non richiede di associare informazione di indirizzamento ai dati trasmessi. È una commutazione di tipo *posizionale:* una volta creato il circuito, si sa che l'informazione che arriva ad un commutatore da un dato circuito di ingresso deve essere portata ad un dato circuito di uscita. Con la commutazione di pacchetto, invece, dobbiamo associare ai pacchetti dell'informazione di indirizzamento, in modo da dire al nodo di commutazione come instradare il pacchetto. In assenza di circuiti virtuali, ogni pacchetto porta un'identità univoca (indirizzo) del destinatario. Si ha una commutazione basata su identificatori (o etichette: *label switching*). In presenza di circuiti virtuali, possono essere utilizzati identificatori di circuito virtuale in ogni pacchetto. Tali identificatori hanno sovente significato locale alla singola tratta trasmissiva, sono allocati al momento della creazione del circuito virtuale, e vengono memorizzati nei nodi di commutazione come informazione ausiliaria (di stato) sul circuito virtuale: quando al nodo x lungo il percorso tra la sorgente A e la destinazione B si crea il circuito virtuale, si decide l'instradamento (cioè quale è la porta p_{xB} su cui instradare i pacchetti del circuito virtuale, e si alloca una etichetta (o identificatore di circuito virtuale) l_{xAB} sul canale attestato in p_{xB}. Al successivo nodo y lungo il percorso vengono svolte le stesse operazioni: i pacchetti arriveranno sulla porta p_{yA} e verranno instradati verso la porta p_{yB}, allocando l'etichetta l_{yAB} sulla tratta successiva, e memorizzando che un pacchetto all'ingresso p_{yA} con etichetta l_{xAB} deve essere portato all'uscita p_{yB}, cambiando l'etichetta da l_{xAB} a l_{yAB} (operazione di *label swapping*). Nei pacchetti dati non sarà più necessario memorizzare la piena identità del destinatario: è sufficiente memorizzare le etichette che, avendo solo significato locale, tipicamente possono essere di meno rispetto al numero di indirizzi (che hanno significato globale), quindi occupano meno spazio nelle intestazioni del pacchetti.

1.6 Topologie di rete

Adottando un punto di vista molto astratto, le reti di telecomunicazioni possono essere rappresentate mediante grafi in cui le centrali di commutazione (o nodi della rete) sono rappresentate da vertici, mentre i canali di comunicazione sono rappresentati da archi (diretti se i canali sono unidirezionali e non diretti se i canali sono bidirezionali).

Una topologia è quindi rappresentata da un grafo $G = (V,A)$, dove V è un insieme di vertici ed A un insieme di archi non orientati od orientati.

Nel seguito descriviamo brevemente le più comuni topologie di rete, illustrandone in modo molto schematico le principali caratteristiche.

1.6.1 Maglia completamente connessa

Una topologia a *maglia completamente connessa* è rappresentata da un grafo dove ogni vertice è collegato con tutti gli altri tramite un arco bidirezionale (o due archi unidirezionali). Ciò significa che esistono tutte le possibili connessioni tra i vari nodi della rete (si veda la figura 1.5).

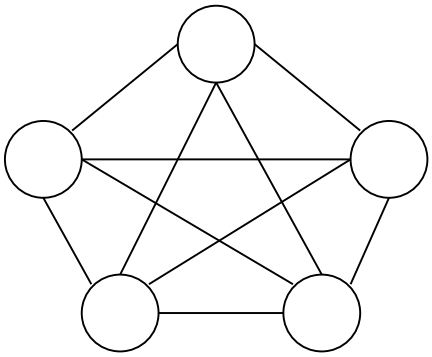


Figura 1.5. Maglia completamente connessa

Indicando con $|V|$ la cardinalità dell'insieme dei vertici del grafo e con $|A|$ la cardinalità dell'insieme degli archi (bidirezionali), in questo caso, ponendo $|V| = N$, si ha:

$$|A| = \binom{N}{2} = \frac{N(N-1)}{2} = \frac{N^2 - N}{2} \tag{1.1}$$

Da tale espressione si nota che la crescita del numero di canali in una topologia completamente connessa è grosso modo proporzionale al quadrato del numero dei nodi. Questo ci fa capire che è insensato pensare ad una topologia di questo tipo per una rete medio-grande, proprio per il fatto che il numero di collegamenti tra i vari nodi è troppo elevato. Già nel caso di una rete con mille nodi (cioè una rete di dimensioni non esagerate) sono necessari circa un milione collegamenti. Per tale motivo questa topologia è usata solo in casi molto particolari, quando il numero di nodi è molto basso.

La topologia a maglia completamente connessa offre però un significativo vantaggio: in questo caso non esistono praticamente problemi di commutazione, in quanto la costruzione del canale che collega due interlocutori qualsiasi si risolve andando a scegliere quel particolare canale che collega i due utenti. Inoltre, la presenza di un così alto numero di canali rende la topologia molto adatta a tollerare la presenza temporanea di eventuali guasti.

1.6.2 Albero

Eliminando da una topologia a maglia completamente connessa tutti i canali che non sono indispensabili per permettere ai nodi di comunicare, si arriva ad una topologia ad *albero* (si veda la figura 1.6).

Questa topologia è caratterizzata dal fatto che $|A| = N - 1$. Ciò comporta degli ovvii svantaggi. In questo caso tra due nodi qualsiasi esiste un unico percorso fisico, quindi se un canale si satura o si guasta la rete non è più in grado di funzionare proficuamente.

I vantaggi della topologia ad albero derivano dalla semplicità della topologia e nella commutazione, perchè l'esistenza di un unico cammino tra due nodi rende molto semplici le procedure per la costruzione di un collegamento tra due utenti. Il ridotto numero di canali implica anche bassi costi.

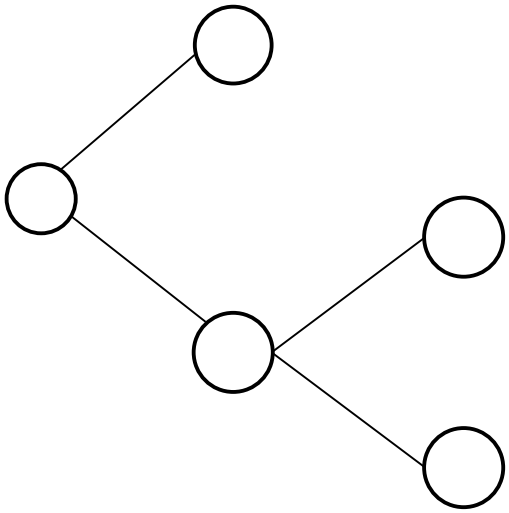


Figura 1.6. Albero

1.6.3 Maglia non completamente connessa

Questa topologia, o meglio questa classe di topologie, detta anche semplicemente *maglia*, raggruppa un insieme di topologie di reti nelle quali $|A|$ è maggiore del numero minimo di archi necessari a collegare tutti i nodi tramite un albero, cioè $N-1$, ma minore del massimo dato dalla (1.1):

$$N-1 < |A| < \binom{N}{2} \tag{1.2}$$

Rispetto alla topologia a maglia completamente connessa, $|A|$ è diminuita da una dipendenza dal quadrato del numero di nodi ad un valore che al limite inferiore è dell'ordine del numero dei nodi.

Per non incorrere negli svantaggi propri della topologia ad albero, si costruiscono reti a maglia che hanno un numero di canali tale da disporre di cammini alternativi tra due nodi. Nell'esempio della figura 1.7, due nodi qualsiasi possono essere collegati anche in presenza di un guasto.

La topologia a maglia viene usata in molte classi di reti di telecomunicazioni. Rispetto alla topologia a maglia completamente connessa, si ha il vantaggio di un minor numero di canali, ma lo svantaggio di una maggiore difficoltà nella commutazione, dovuto al calcolo dell' instradamento. Si noti che questa topologia consente di adattare la rete alla reale dislocazione geografica dei nodi da interconnettere.

1.6.4 Topologie regolari

Esistono topologie regolari che hanno caratteristiche interessanti e che sono usate per particolari classi di reti di telecomunicazioni. In particolare consideriamo topologie a *stella*, ad *anello*, e a *bus*.

Stella

La topologia a stella è raffigurata nella figura 1.8. Tutti i nodi della rete sono collegati ad un elemento particolare detto *centro stella* attraverso il quale transita tutto il traffico della rete.

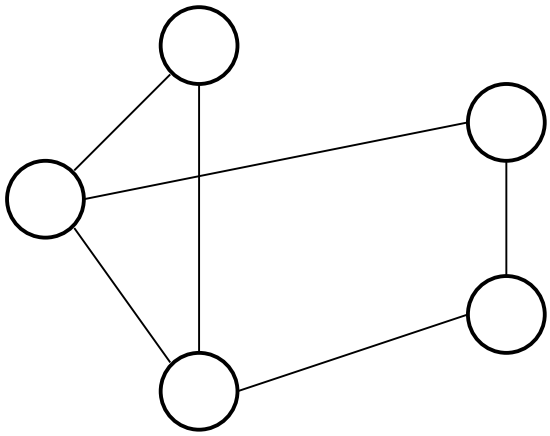


Figura 1.7. Maglia biconnessa: ogni nodo è connesso a due canali

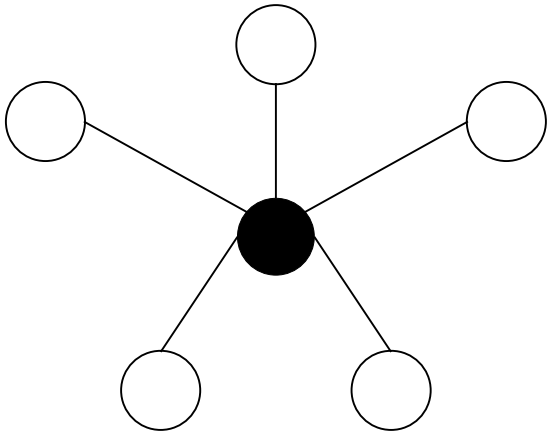


Figura 1.8. Stella

Il grafo che rappresenta la rete è caratterizzato dall'avere $|V| = N + 1$ ed $|A| = N$. Anche in questo caso il numero di canali che collegano i vari nodi è quindi prossimo al minimo valore possibile.

Dal punto di vista del nodo, la commutazione risulta molto semplice in quanto tutto il traffico deve essere diretto al centro stella.

Dal punto di vista del centro stella esistono due possibilità. Nel caso di centro stella *attivo*, il centro stella svolge la maggior parte delle funzioni di commutazione della rete e deve quindi essere opportunamente dimensionato. Nel caso di centro stella *passivo*, la rete è a diffusione, nel senso che tutte le trasmissioni vengono inviate dal centro stella ad ogni nodo, indipendentemente dalla loro destinazione, senza svolgere operazioni di commutazione.

In entrambi i casi esistono notevoli problemi legati all'affidabilità, che sono più gravi nel caso di un centro stella attivo, in quanto un elemento attivo di notevole complessità si guasta più facilmente di un elemento passivo. È da notare che un guasto al centro stella rende inoperante tutta la rete.

La topologia a stella è estremamente diffusa nelle reti di telecomunicazioni. È stata la prima configu-

razione delle reti telefoniche, dove i terminali d'utente si collegano in modo stellare alla loro centrale di commutazione. È caratteristica di reti su canale radio e via satellite, nelle quali i terminali, soventi dotati di mobilità, colloquiano con una stazione fissa o con il satellite. Essa è ultimamente stata usata anche nelle LAN, originariamente concepite per altre topologie regolari (bus e anello).

Anello

Nella topologia ad anello si utilizza un cammino circolare chiuso per collegare tutti i nodi (si veda la figura 1.9).

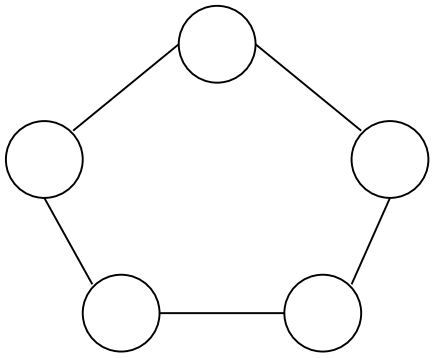


Figura 1.9. Anello

Le caratteristiche del grafo sono $|V| = N$, $|A| = N$.

I canali possono essere o bidirezionali o unidirezionali. Si noti che questo è l'unico caso in cui N canali *unidirezionali* sono sufficienti per collegare N utenti. Nei casi dell'albero e della stella i canali dovevano essere necessariamente bidirezionali. Con canali unidirezionali l'informazione ha a disposizione un solo cammino per raggiungere la destinazione, mentre con canali bidirezionali i cammini sono due. In entrambi i casi le operazioni relative alla commutazione sono semplici.

Lo svantaggio più evidente della topologia ad anello è dovuto alla sua ridotta affidabilità: nel caso di anello unidirezionale basta un guasto ad interrompere la rete, mentre la topologia ad anello bidirezionale può essere riconfigurata in una topologia ad anello unidirezionale in caso di un guasto isolato. Un esempio di riconfigurazione nel caso di un guasto di un nodo è illustrato nella figura 1.10.

La topologia ad anello è stata usata per alcune tipologie di reti locali di calcolatori. Attualmente viene molto impiegata nelle dorsali di reti pubbliche utilizzanti gli schemi di multiplazione temporale SDH (Synchronous Digital Hierarchy) o SONET sui collegamenti, che prevedono copertura ad anelli interconnessi per motivi di protezione da situazioni di guasto.

Bus

La topologia a Bus realizza il collegamento tra i nodi della rete usando un unico canale. Ovviamente tale canale deve essere collegato ad ogni nodo sia in trasmissione sia in ricezione (si veda la figura 1.11). Si usa quindi un canale broadcast (ad accesso multiplo e a diffusione circolare) su cui tutti possono trasmettere e da cui tutti possono ricevere informazione.

La topologia porta all'implementazione di una rete a diffusione bidirezionale. Per ottenere la selezione dell'interlocutore desiderato è necessario che i ricevitori operino un filtraggio dell'informazione ricevuta sulla base di un indirizzo (implicito od esplicito).

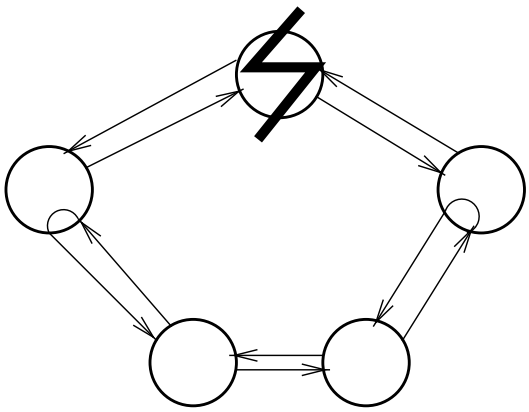


Figura 1.10. Riconfigurazione di un anello bidirezionale nel caso di un guasto ad un nodo

Si noti la somiglianza tra la topologia a bus e quella a stella passiva: il bus funziona come un centro stella passivo distribuito nello spazio.

L'implementazione della topologia a bus può prevedere accoppiamenti passivi o attivi dei nodi al bus, come mostrato nella figura 1.11. Nel secondo caso, se l'informazione che si propaga nelle due direzioni non è la stessa, si parla di *doppio bus*.

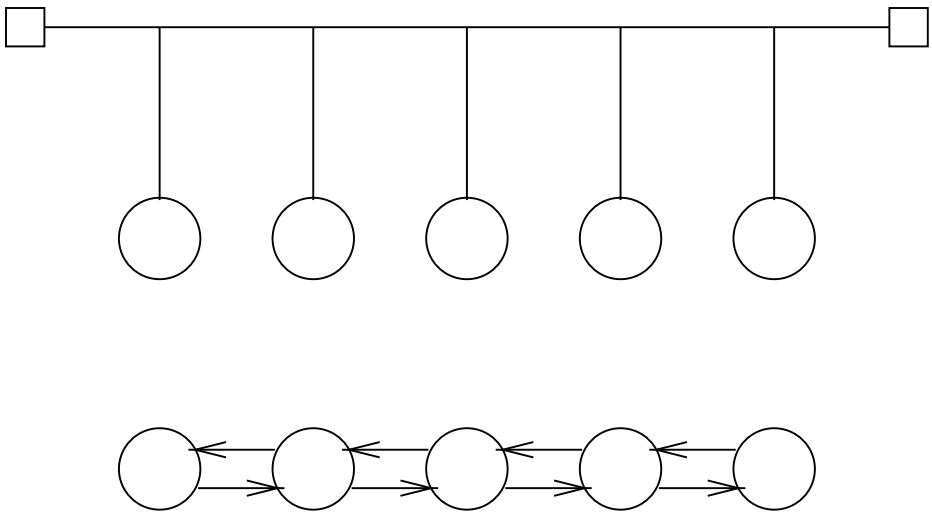


Figura 1.11. Bus

La topologia a bus è usata in particolar modo per reti locali di calcolatori.

1.6.5 Topologie miste

Spesso si incontrano nella pratica reti con topologia mista. Le categorie principali di topologie miste sono la *topologia gerarchica* e la *topologia ad interconnessione*.

Topologia gerarchica

In una rete con topologia gerarchica si hanno nodi che hanno la funzione di raccogliere il traffico di una porzione della rete per portarlo al livello gerarchico superiore (si veda la figura 1.12).

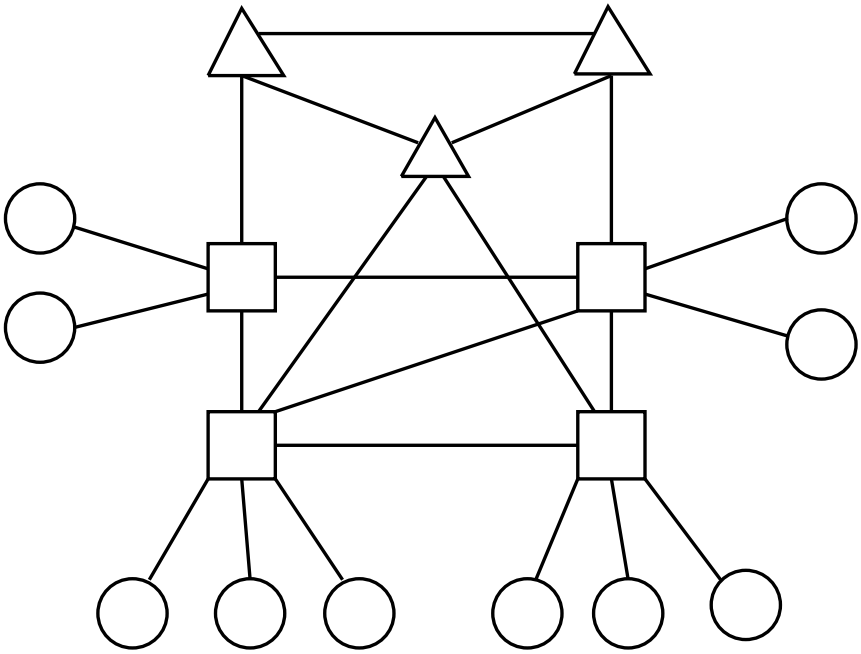


Figura 1.12. Topologia gerarchica

Un esempio è dato dalla rete telefonica dove al livello più basso frequentemente si usa una topologia a stella, ai livelli intermedi si adotta una topologia a maglia parzialmente connessa e al livello più alto si impiegano frequentemente topologie a maglia completamente connessa.

Un altro esempio è fornito da alcune reti locali che prevedono due livelli di gerarchia. Per esempio, in alcuni casi si utilizza un primo livello con topologia ad anello unidirezionale, ed un secondo livello con topologia ad anello bidirezionale oppure con topologia a Bus.

Topologia ad interconnessione

Sempre più frequentemente si tende a costruire reti di reti, collegando reti eterogenee: questo è il paradigma fondamentale della rete Internet. In questo caso si ottengono reti con topologie derivanti dall'interconnessione delle topologie delle reti componenti.

Gli elementi di interconnessione utilizzati possono prendere nomi diversi: *switch*, *bridge*, *brouter*, *router* o *gateway*. La distinzione tra i vari tipi di elementi di interconnessione è data dalle funzioni necessarie per collegare le reti, quindi dalla differenza tra le reti collegate.

Un esempio frequente è fornito da reti locali con topologia a bus collegate tramite bridge ad una rete più veloce con topologia ad anello (si veda la figura 1.13). Internet prevede sottoreti di topologia arbitraria interconnesse attraverso router che eseguono il protocollo IP (Internet Protocol).

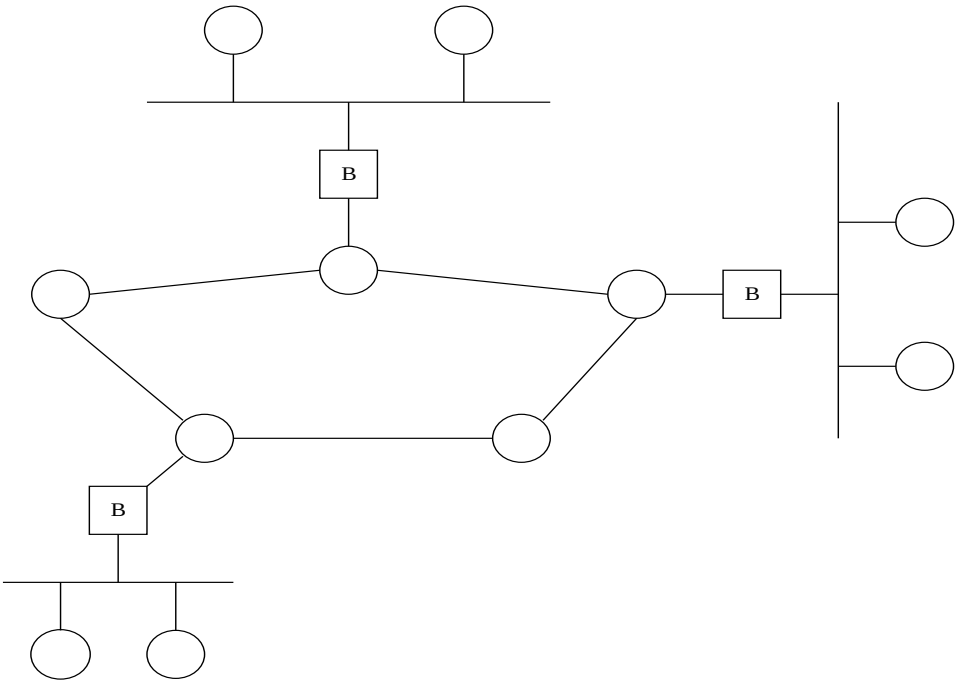


Figura 1.13. Topologia ad interconnessione

1.7 Reti telefoniche

La rete telefonica è una rete che offre servizi di telefonia tradizionali ed avanzati ad una vastissima popolazione di utenti.

Il servizio base offerto dalla rete telefonica è il trasferimento del segnale vocale (sempre più spesso in forma numerica) tra due interlocutori che vengono collegati su richiesta di uno di essi.

Il collegamento tra due utenti segue le seguenti fasi:

- impegno: l'utente chiamante segnala alla rete la sua intenzione di avviare le procedure per l'apertura di un collegamento;
- selezione: l'utente chiamante fornisce alla rete il numero identificatore dell'utente chiamato;
- instradamento: la rete determina un opportuno percorso per il collegamento dell'utente chiamante all'utente chiamato; viene così costruito un circuito per la comunicazione; la costruzione del circuito può fallire per la mancanza di risorse interne alla rete (blocco);

- conversazione: la rete trasferisce in modo bidirezionale i dati risultanti dalla numerizzazione del segnale vocale dei due utenti;
- svincolo: su richiesta di uno dei due utenti, la rete rilascia le risorse impegnate per la costruzione del circuito.

La rete telefonica ha normalmente una topologia gerarchica del tipo di quella illustrata nella figura 1.14.

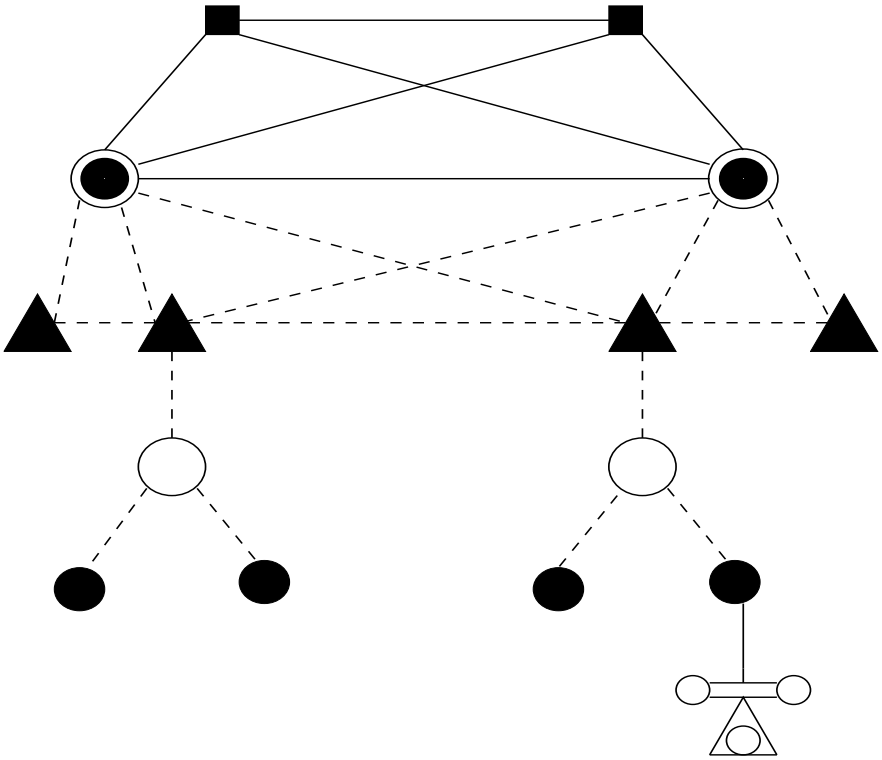


Figura 1.14. Topologia di rete telefonica

La topologia della parte periferica della rete è una stella, in quanto i vari apparati di utente devono essere collegati ad una centrale di commutazione locale. Le centrali di commutazione ai vari livelli gerarchici sono poi collegate tramite reti a maglia, finchè al massimo livello della gerarchia normalmente esiste una rete a maglia completamente connessa.

La rete telefonica italiana è articolata in quattro livelli gerarchici, oltre al livello di collegamento diretto degli apparati di utente alle centrali locali (centrali urbane).

La rete *primaria* collega i tre centri nazionali di Milano Roma e Palermo ai 21 centri di compartimento. I tre centri nazionali sono collegati con una maglia completamente connessa, mentre non tutti i centri di compartimento sono collegati direttamente.

Gli altri tre livelli gerarchici sono costituiti dalla rete *compartimentale*, che collega i 233 centri di distretto ai centri di compartimento, dalla rete *distrettuale*, che collega i 1399 centri di settore ai centri di distretto, e dalla rete *settoriale* che collega le centrali urbane ai centri di settore.

Tutte e tre queste reti hanno una topologia prossima ad una stella (vi sono alcuni collegamenti trasversali in aggiunta ad una topologia a stella).

La stratificazione nei quattro livelli è illustrata dalla figura 1.15. Attualmente tale schema è in fase di revisione, in quanto la maggior capacità delle centrali consente la riduzione dei livelli gerarchici.

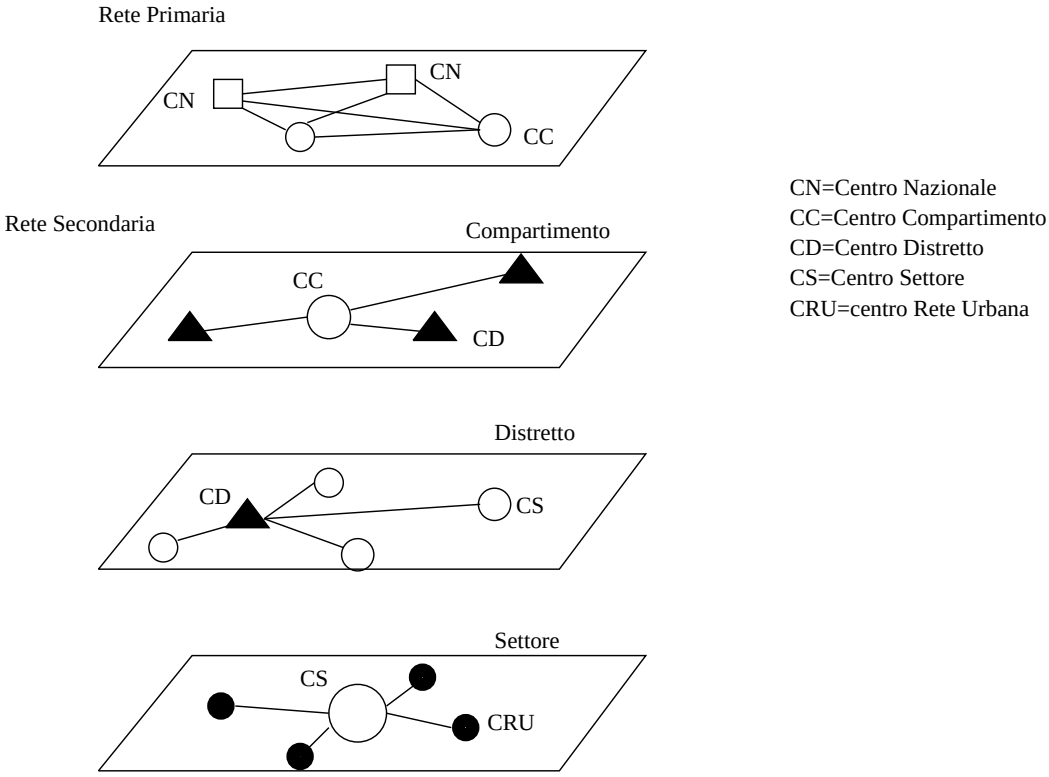


Figura 1.15. Architettura della rete telefonica italiana

Nella fase dell'instradamento la rete deve scegliere quali collegamenti impegnare per instaurare un circuito tra l'utente chiamante e l'utente chiamato. Con riferimento al modello topologico della rete ciò significa scegliere un cammino che colleghi il nodo di partenza al nodo di destinazione. La scelta nel caso della rete telefonica viene effettuata in modo deterministico, secondo criteri che indicano le priorità di scelta tra le varie alternative possibili.

Normalmente l'instradamento viene effettuato sezione per sezione, cioè la ricerca viene effettuata dalla centrale locale ad una prima centrale di transito e poi si demanda la prosecuzione dell'instradamento a quest'ultima, finchè non viene raggiunta la centrale locale di destinazione.

1.8 Reti telematiche

Il disegno nella figura 1.16 rappresenta una rete di calcolatori. Il significato dei simboli è il seguente

H	=	HOST (elaboratore utente)
T	=	TERMINALE
∘	=	NODO
——	=	CANALE
P	=	PAD
LAN	=	LOCAL AREA NETWORK (rete locale dell'utente)
MAN	=	METROPOLITAN AREA NETWORK (rete metropolitana pubblica)
G	=	GATEWAY
R	=	ROUTER

Con il termine "HOST" indichiamo un elaboratore con una sua periferia di terminali, collegato alla rete di cui usa i servizi, eventualmente per fornire servizi a valore aggiunto ad altri utenti.

Con l'acronimo "PAD" (Packed Assembler/Disassembler) si indica un'apparecchiatura di interfaccia tra un terminale che funziona in modo asincrono ed un nodo a commutazione di pacchetto.

Il "GATEWAY" e il "ROUTER" sono elementi intelligente che permettono la interconnessione tra due tipi di reti differenti.

Le "LAN" sono reti locali di calcolatori private che vengono usate per collegare gli elaboratori ed i terminali interni ad una stessa società.

Le "MAN" sono reti metropolitane pubbliche che vengono usate per fornire servizi di trasferimento dati ad alta velocità e per raccogliere traffico da inviare sulle reti a lunga distanza.

La topologia di una rete di calcolatori a larga estensione geografica (WAN – wide area network) è tipicamente una maglia scarsamente connessa. Nel caso di reti metropolitane (MAN) e ancor più di reti locali (LAN) si tende invece ad utilizzare topologie regolari.

La tecnica di commutazione usata nelle reti di calcolatori è la commutazione di pacchetto, sovente combinata con l'utilizzo di circuiti virtuali per garantire buone caratteristiche al trasferimento dell'informazioni.

1.9 Modelli

La costruzione di modelli matematici astratti per lo studio quantitativo delle reti di telecomunicazioni è di fondamentale importanza per risolvere due problemi:

- l'analisi delle prestazioni di una prefissata configurazione di rete a fronte di un dato livello di carico, definito in funzione del traffico generato dagli utenti della rete e dai servizi da loro richiesti;
- il dimensionamento delle risorse necessarie a soddisfare un dato insieme di richieste di servizio (e quindi a sopportare un dato livello di traffico) rispettando vincoli di qualità predefiniti.

I modelli matematici sviluppati per affrontare e risolvere i due problemi citati costituiscono la *teoria del teletraffico*. Questa si basa su modelli probabilistici dove

- l'andamento temporale delle richieste di servizio da parte degli utenti viene descritto mediante processi stocastici opportuni,
- la quantificazione del servizio viene fornita mediante variabili aleatorie opportune.

A seconda del fatto che le richieste prodotte dagli utenti possano essere fatte attendere o meno prima del soddisfacimento, la teoria del teletraffico considera

- sistemi a perdita,

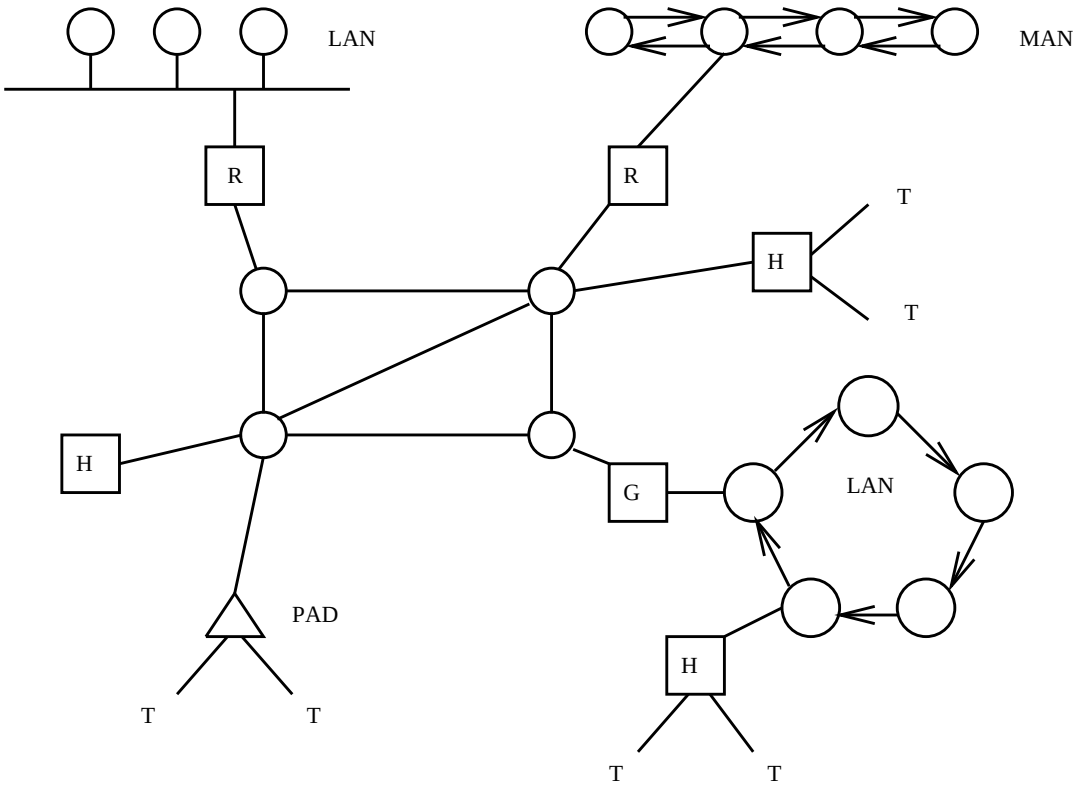


Figura 1.16. Rete di calcolatori

- sistemi a coda.

Mentre lo studio dei sistemi a perdita ha costituito la base per l'analisi ed il progetto delle reti telefoniche (e delle reti a circuito in generale) nelle loro prime fasi di sviluppo, oggi la teoria delle code e quindi lo studio dei sistemi a coda sta assumendo una importanza sempre maggiore anche nello studio delle reti a commutazione di circuito.

Nel caso di reti telematiche a commutazione di messaggio o di pacchetto la teoria delle code, ed in particolare lo studio delle *reti di code*, fornisce lo strumento principale di analisi e di progetto.

Ciò è facilmente comprensibile anche da un modello estremamente semplice ed astratto del funzionamento di un commutatore di messaggio o di pacchetto, quale quello illustrato nella figura 1.17.

Dai canali (nella figura per comodità sono stati disegnati separatamente i canali unidirezionali uscenti ed entranti) entrano nel nodo i pacchetti che sono memorizzati in buffer opportunamente predisposti.

I pacchetti sono poi processati da un elaboratore (elemento fondamentale del nodo), il quale verifica la correttezza dell'informazione ricevuta e calcola l'instradamento verso la destinazione. In base al risultato ottenuto i pacchetti sono messi in coda per la trasmissione su uno dei canali uscenti.

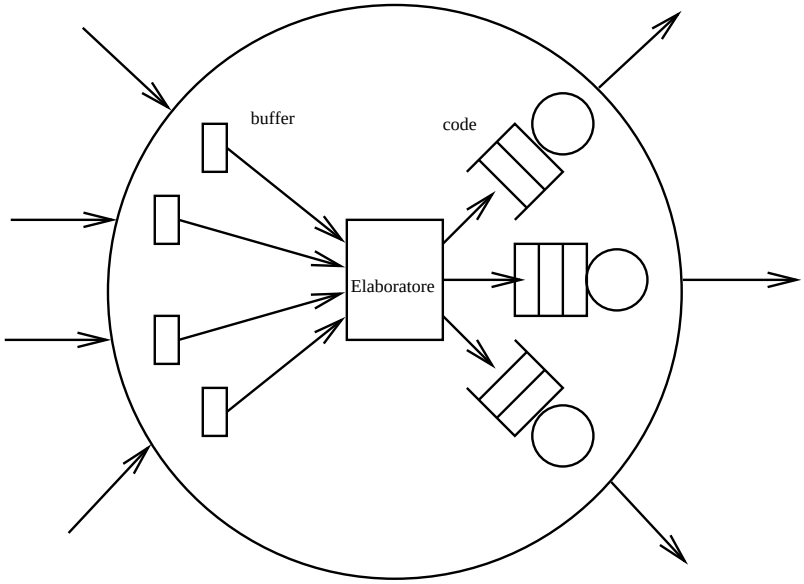


Figura 1.17. Modello di commutatore di pacchetto

La trasmissione può dover essere differita a causa dell'impegno del canale prescelto. È quindi necessario associare una *coda* di pacchetti ad ognuna delle uscite dal nodo.

Se ora immaginiamo di collegare tra loro più nodi di una rete, risulta evidente il fatto che si ottiene un modello composto da una rete di code interconnesse.

Lo schema che utilizzeremo per rappresentare una coda è quello illustrato nella figura 1.18. Si possono immediatamente distinguere la fila di attesa e la stazione di servizio, che può comprendere uno o più servitori. Si notano inoltre gli arrivi alla fila di attesa e le partenza dei clienti dalla stazione di servizio.

Nei capitoli che seguono verranno presentati alcuni risultati elementari della teoria delle code utili per lo studio delle reti di telecomunicazioni, sia a commutazione di circuito, sia a commutazione di pacchetto.

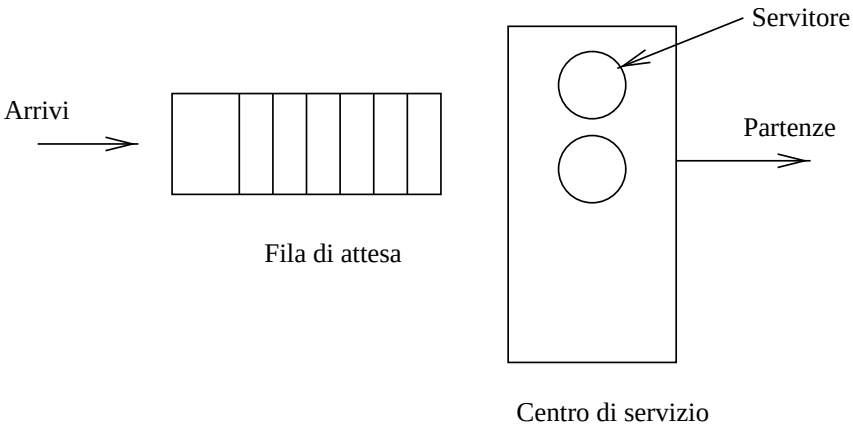


Figura 1.18. Rappresentazione di una Coda

2

Simulazione

2.1 Analisi e simulazione a confronto

Per valutare le prestazioni di un sistema dinamico, caratterizzato da uno stato e da un insieme di componenti che interagiscono, esistono due approcci sostanzialmente differenti: l'**analisi** e la **simulazione**. Entrambi gli approcci, a causa delle grandi dimensioni dei sistemi che sovente si vogliono studiare, richiedono la costruzione di un buon **modello** che leghi i parametri di ingresso agli indici di prestazione che si vogliono osservare. Un buon modello deve riprodurre bene il funzionamento del sistema reale che si vuole studiare, ma nello stesso tempo deve presentare una complessità accettabile. Mentre l'approccio analitico richiede un grande sforzo di astrazione per arrivare al modello, ma è molto rapido nel valutare gli indici di prestazione (attraverso soluzioni analitiche o numeriche), l'approccio simulativo, al contrario, costa meno nella costruzione del modello, ma necessita di lunghi periodi tempo per l'esecuzione ed il calcolo statistico delle prestazioni. Inoltre produce stime degli indici di prestazione affette da incertezza statistica.

2.1.1 Pianificazione e realizzazione di una simulazione

La pianificazione di una simulazione consta di molteplici passi. A grandi linee, si possono individuare i seguenti passi.

1. **Formulazione del problema**. Questa fase non presenta particolari problemi di carattere statistico ma, comportando la definizione dei risultati della simulazione, condiziona tutte le altre. In generale questa fase serve a determinare le relazioni funzionali tra ingressi ed uscite (indici di prestazione).
2. **Raccolta ed analisi dei dati**. Si tratta essenzialmente di raccogliere dati sperimentali sulle distribuzioni delle grandezze aleatorie che intervengono (come dati d'ingresso) nella simulazione.
3. **Formulazione del modello funzionale o matematico**. Questo è il passo più delicato in quanto è in questa fase che si decide come semplificare il sistema reale in modo da rappresentarlo sul calcolatore.
4. **Scrittura del programma di simulazione**.
5. **Valutazione del modello di simulazione**. Si confrontano le uscite del modello di simulazione con quelle del sistema reale. Si può ad esempio verificare, con un test statistico, possibilmente a parità di dati di ingresso, se lo scarto tra le prestazioni reali del sistema e quelle della simulazione sia significativo oppure no. Se non è disponibile un sistema reale, si può verificare ciò simulando un caso particolarmente semplice per cui esista una soluzione analitica.

6. **Analisi dei risultati**. Raccolta ed interpretazione dei risultati delle simulazioni.

2.2 Simulatori orientati agli eventi e orientati ai processi

I simulatori possono essere suddivisi in due grandi categorie: **simulatori sincroni** e **asincroni**.

Nei simulatori sincroni il tempo di simulazione viene diviso in tanti intervalli di ugual ampiezza ed alla fine di ciascuno di questi si determinano gli eventuali cambiamenti di stato del sistema. Nei simulatori asincroni, il sistema cambia stato in certi momenti e la simulazione *salta* da un istante di cambiamento di stato all'altro. Questa di solito è la tecnica più conveniente per simulare sistemi di servizio (ad esempio una coda).

Un'ulteriore suddivisione può essere fatta tra simulatori *orientati agli eventi* e *orientati ai processi*.

Le simulazioni ad *eventi discreti* sono quelle nelle quali si gestiscono individualmente i cambiamenti di stato del sistema, detti *eventi*, che non avvengono in modo continuo, ma si producono in istanti temporali (aleatori) discreti. Tipico esempio sono le simulazioni dei sistemi di servizio, dove gli eventi corrispondono o ad un arrivo nel sistema o ad un fine servizio. Le simulazioni in cui il cambiamento di stato avviene in modo continuo si definiscono ad eventi continui.

Un simulatore orientato agli eventi di solito utilizza un linguaggio di programmazione *general purpose*. Quando, durante la simulazione, l'attenzione non è rivolta ai singoli eventi ma ai "processi", che rappresentano la dinamica dei sottosistemi si parla di simulatori orientati ai processi. I processi eseguono le loro attività e interagiscono tra di loro. Normalmente un simulatore a processi viene costruito utilizzando o un linguaggio di programmazione non *general purpose*, o un pacchetto software per la simulazione e per la gestione e l'implementazione di processi concatenati, come Bones e Opnet.

Durante una simulazione si possono osservare due fasi: una fase di transitorio iniziale ed una fase di regime. Durante la prima fase, i dati raccolti dalla simulazione sono legati al transitorio del sistema, quando le condizioni iniziali influenzano in modo significativo le prestazioni. Quando il comportamento del sistema non dipende più dalle condizioni iniziali con cui la simulazione parte e le probabilità di stato sono indipendenti dal tempo si parla di fase di regime; il sistema in questa fase ha raggiunto la condizione di equilibrio statistico (qualora questo esista).

2.2.1 Esempio: la coda M/M/1

Approccio simulativo (simulatore ad eventi discreti sincrono)

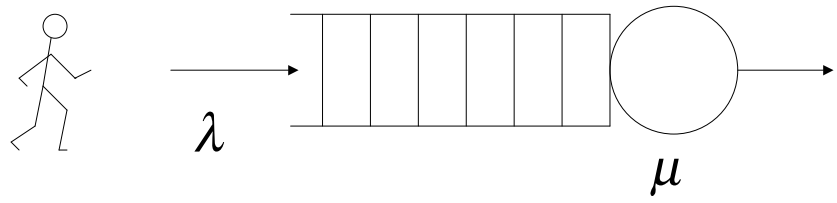


Figura 2.1. Un sistema a coda M/M/1

Una coda M/M/1 è un sistema che prevede la presenza di un servitore, tempi di arrivo dei clienti nel sistema distribuiti esponenzialmente e tempi di servizio distribuiti esponenzialmente (si veda la figura 2.1).

Le variabili utilizzate dal simulatore sono:

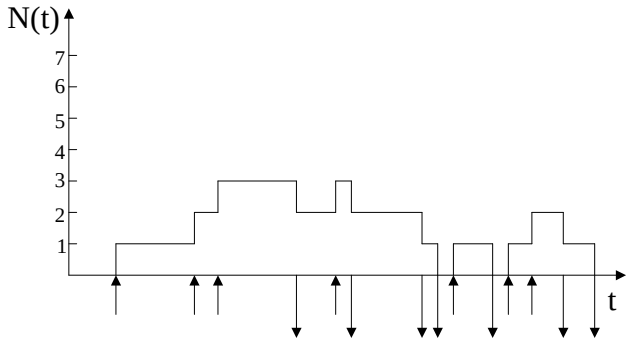
- variabili di ingresso: λ (tasso dei tempi di arrivo dei clienti alla coda) e μ (tasso dei tempi di servizio);
- variabili di stato: N (numero di clienti nella coda);

Le variabili di uscita osservate possono essere:

- numero medio di clienti nella coda, $E[N]$;
- tempo medio di attesa di un cliente nel sistema, $E[W]$.

Definite le variabili in gioco, si supponga di conoscere (per esempio avendo effettuato delle misure), una sequenza di istanti di arrivo $A[k]$ di clienti e dei corrispondenti tempi di servizio $S[k]$, dove $S[k]$ è la durata del servizio del cliente k arrivato al tempo $A[k]$. Si decide un passo di discretizzazione Δt dell'asse dei tempi e si costruisce un simulatore sincrono.

Scandendo i vettori $A[k]$ e $S[k]$ si può costruire un vettore $N[n]$ che indica il numero di clienti presenti nel sistema nell'intervallo di tempo $[(n-1)\Delta t, n\Delta t]$. Dal vettore $N[n]$ si possono ottenere le variabili di uscita $E[N]$ e $E[W]$ (si veda la figura 2.2).



$$E[N] = \frac{\text{Integrale di N(t)}}{\text{Tempo totale di simulazione}}$$

$$E[W] = \frac{\text{Integrale di N(t)}}{\text{Numero di partenze}}$$

Figura 2.2. Sequenza degli arrivi e delle partenze in un sistema a coda M/M/1

I problemi legati a questa simulazione sincrona della coda, sono essenzialmente:

- la definizione del passo di discretizzazione dell'asse dei tempi;
- perdita di precisione per effetto della discretizzazione;
- perdita di efficienza per effetto della discretizzazione.

Approccio analitico

Un diverso approccio per valutare le stesse variabili di uscita è quello analitico. Supponendo che i tempi di arrivo e di servizio siano distribuiti esponenzialmente, le grandezze si possono facilmente determinare, con semplici concetti di base di teoria delle code, come:

$E[N] = \frac{\rho}{1-\rho}$ dove $\rho = \frac{\lambda}{\mu}$;

$E[W] = \frac{1}{\mu - \lambda}$.

2.3 Simulazione ad eventi

Il modello di simulazione reagisce agli eventi mettendo in calendario nuovi (futuri) eventi. L'esecuzione è quasi parallela.

Esistono diversi concetti di tempo:

- **tempo reale**;
- **tempo simulato**;
- **tempo di esecuzione**.

Sono necessari:

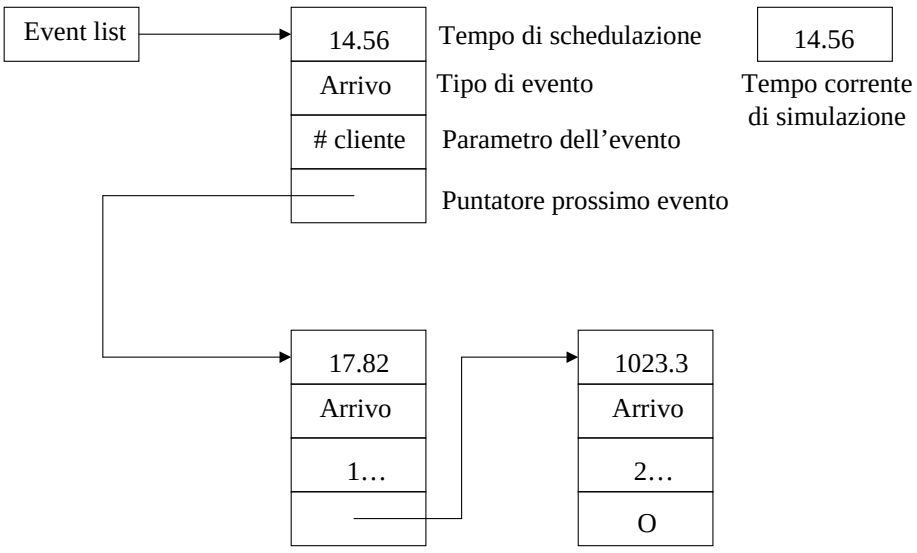


Figura 2.3. Modello di un simulatore ad eventi

- un calendario degli eventi che sono in attesa di ricevere (**event list**);
- una struttura dati per descrivere gli eventi (**event notice** o **event record**) (si veda la figura 2.3).

2.4 Generazione di sequenze pseudocasuali

I generatori di numeri casuali sono basati su generatori di sequenze pseudo-casuali.

"Una sequenza casuale è una vaga nozione soggiacente all'idea di una sequenza nella quale ogni termine è impredittibile ai non addetti ai lavori, ed i cui numeri superano dei test, tradizionali per gli statistici, e indipendenti comunque dall'uso al quale la sequenza è destinata" (Lemher, 1951).

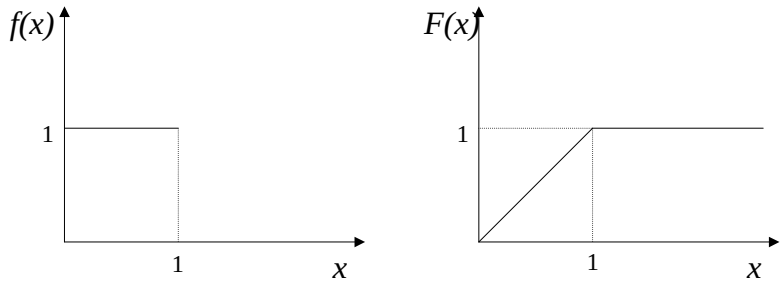


Figura 2.4. Densità di probabilità uniforme e sua distribuzione cumulativa

I requisiti che normalmente si pongono ad un generatore di sequenze pseudocasuali sono:

- ripetibilità;
- soddisfacimento dei test;
- semplicità e rapidità;
- periodicità lunga;
- portabilità.

Di solito si costruiscono generatori di sequenze di numeri indipendenti e uniformemente distribuiti tra 0 e 1, caratterizzati come mostrato nella figura 2.4.

L'algoritmo utilizzato comunemente per la generazione dei numeri pseudocasuali è basato sull'operazione di modulo.

Una sequenza di numeri interi è generata da:

$$x_{n+1} = x_n \times a \bmod m$$

dove x_0 è detto seme (iniziale) della sequenza. Se $x_0 \neq 0$, per le proprietà dell'operazione di modulo, i numeri possibili sono:

$$1,2,\ldots,m-2,m-1$$

quindi la sequenza generata ha una periodicità massima pari a $m-1$. Per questo motivo m è anche detto *periodo della sequenza*. Si fa presente che m deve essere un numero primo per garantire massima periodicità, ovvero che la sequenza generata contenga in un periodo una sola volta tutti gli $m-1$ numeri escluso lo 0.
Qui bisognerebbe definire che cosa è un numero primitivo.

Esempio 1

Con $a=5$, $m=7$, e $x_0=1$ si genera la sequenza: 1, 5, 4, 6, 2, 3, 1. In questo modo si generano tutti i numeri compresi tra 1 e 6 con una periodicità limitata a causa della scelta di a ed m.

Esempio 2

Con $a=7$, $m=11$, e $x_0=1$ si genera la sequenza: 1, 7, 5, 2, 3, 10, 4, 6, 9, 8, 1. In questo modo si generano tutti i numeri compresi tra 1 e 10 con una periodicità limitata a causa della scelta di a ed m.
Metterei anche un esempio con periodo più corto.

2.4.1 Teorema di Fermat

La funzione $\varphi()$ di Eulero conta il numero di numeri primi con il numero ad argomento. Esempio: $\varphi(p)=p-1$ se p è un numero primo.

Un teorema dovuto a Fermat dice che: $a^{\varphi(p)}=1 \bmod p$, $\forall p$ e $\forall a<p$.

In questo contesto interessa quello che viene detto piccolo teorema Fermat: se p è un numero primo, $a^{\varphi(p)}=a^{p-1}=1 \bmod p$.

Dimostrazione del piccolo teorema di Fermat

I numeri

$$a,2a,3a,\ldots,(p-1) \bmod p$$

sono tutti diversi tra loro, quindi sono tutti i numeri compresi tra 1 e $p-1$. Infatti, se per assurdo $na=ma \bmod p$ con $n \neq m$, allora $(n-m)a=ka=0 \bmod p$, con $k=n-m \bmod$ p, ma ciò non è possibile essendo p primo. I $p-1$ numeri distinti possono essere scritti come:

$$1,2,3,\ldots,(p-1) \bmod p$$

uguagliando e semplificando si ottiene l'asserto.

Si deve a questo punto notare che:

- se a è primitivo, $k=p-1$ è il minimo esponente per cui $a^k=1$. Quindi la sequena generata vale:

  $$x_0 a,x_0 a^2,x_0 a^3,\ldots,x_0 a^{p-1}=x_0 \bmod p$$

  dove x_0 è il seme iniziale.

- se a non è primitivo, il generatore ha periodicità pari ad uno dei fattori di $p-1$, visto che esattamente dopo $p-1$ generazioni deve tornare al valore iniziale. I fattori di $p-1$ giocano un ruolo importante sulle proprietà del generatore.

Ad esempio se a è primitivo, a^k è ancora primitivo purché k sia primo con $p-1$. Infatti, se k non è primo con $p-1$, esiste un $l<p-1$ tale per cui $lk=p-1$. Quindi $(a^k)^l=1 \bmod p$ e a^k non è primitivo. Quindi il numero di numeri primitivi con p può essere scritto come $\varphi(p-1)$.

Un buon generatore

Con aritmetica in complemento a due su quattro byte (32 bit) è molto utilizzato il generatore:

$$\begin{aligned} x_{n+1} &= x_n \times a \bmod p \\ p &= 2^{31} - 1 = 2{,}147{,}483{,}647 \\ a &= 7^5 = 16807 \\ p-1 &= 2 \times 3^2 \times 11 \times 31 \times 151 \times 331 \\ x_0 &= 1 \end{aligned}$$

Occorre evitare overflow nella moltiplicazione. Il costo può essere ridotto a quattro moltiplicazioni e due somme.

2.4.2 Generazione di più sequenze indipendenti

Sovente servono più sequenze indipendenti (per esempio due sequenze). Si possono:

- utilizzare due moltiplicatori differenti. Le due sequenze sono indipendenti;
- utilizzare due semi iniziali differenti $x_0^{(1)}$ e $x_0^{(2)}$. I due semi iniziali devono esssere "lontani" nella sequenza generata. Occorrerebbe trovare k e l tali per cui $a^k = x_0^{(1)}$ e $a^l = x_0^{(2)}$. Ciò corrisponde a calcolare dei logaritmi sul campo finito $GF(p)$. La complessità di tale operazione è legata di nuovo alla struttura dei fattori primi $p-1$. Non c'è nessun motivo per non scegliere $x_0^{(1)} = 1$ e $x_0^{(2)} = 2$.
- utilizzare la stessa sequenza generata (con un solo seme iniziale) per avere istanze delle diverse variabili casuali uniformi, partizionando quindi la sequenza in più sequenze pseudocasuali. In linea di principio, non si hanno garanzia di uniformità, ma si conserva la scorrelazione della sequenza originale.

2.5 Metodi per ottenere distribuzioni generali

2.5.1 Metodo della trasformazione inversa

Questo metodo si basa sull'inversione della funzione cumulativa (se questa operazione è possibile).

Sia X una variabile casuale con funzione cumulativa $y = F(x) \in [0,1]$. Se Y è una variabile casuale con distribuzione uniforme si ha:

$$P\{Y \leq y\} = y$$

e anche:

$$P\{Y \leq F(x)\} = F(x)$$

Infatti:

$$P\left\{F^{-1}(Y) \leq F^{-1}(F(x))\right\} = P\left\{F^{-1}(Y) \leq x\right\} = F(x)$$

Per generare una variabile casuale x definita da $F(x)$, si genera una variabile casuale uniforme y e si calcola $F^{-1}(y)$.

Esempi

- Generazione di una variabile casuale positiva con distribuzione esponenziale negativa [si veda la figura 2.5 a)]:

$$\begin{aligned} F(x) &= 1 - e^{-\mu x} \\ y &= 1 - e^{-\mu x} \\ \ln(1-y) &= -\mu x \\ x &= \frac{-\ln(1-y)}{\mu} \end{aligned}$$

- Generazione di una variabile casuale con distribuzione uniforme in (a,b) [si veda la figura 2.5 b)]:

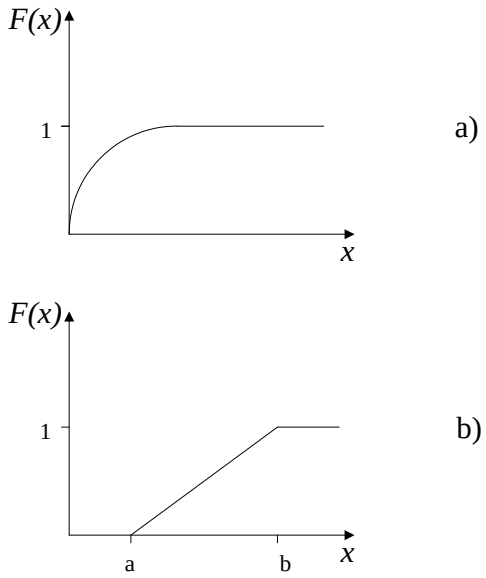


Figura 2.5. Generazione di una variabile casuale con distribuzione esponenziale negativa (a) e con distribuzione uniforme tra (a,b) (b)

$$\begin{aligned} F(x) &= \frac{x-a}{b-a} \\ y &= \frac{x-a}{b-a} \\ x &= (b-a)y + a \end{aligned}$$

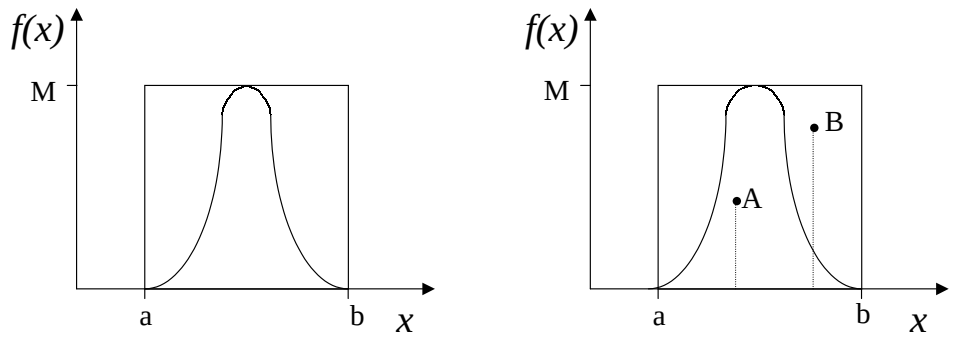


Figura 2.6. Metodo di reiezione

2.5.2 Metodo di reiezione

È un metodo molto utile quando la funzione cumulativa non è nota o non è invertibile. Si generano coppie di numeri casuali (x,y) con x uniforme in (a,b) e y uniforme in $(0,M)$. Se $y \leq f(x)$, si accetta x come istanza della variabile casuale che si vuole generare, altrimenti si genera un'altra coppia di valori (si veda la figura 2.6). La probabilità di accettazione è: $P = 1/[M(b-a)]$, quindi l'efficienza del generatore è $1/[2M(b-a)]$.

2.5.3 Metodo di composizione

È un metodo utilizzato quando la distribuzione è composizione di funzioni note.

Si scompone la funzione $f(x)$ in parti che si generano separatamente:

$$f(x) = \sum_{i=1}^{N} p_i f_i(x) \qquad \sum_{i=1}^{N} p_i = 1$$

Generalizzando questo metodo si può approssimare una densità di probabilità con uno scaloide. Il costo è la generazione di due campioni uniformi (si veda la figura 2.7).

Si noti che:

- Da una variabile casuale uniforme tra 0 e 1, sottraendo 0.5 si ottiene una variabile casuale uniforme tra ± 0.5, con media nulla e varianza $\frac{1}{12}$.
- Se si sommano due variabili casuali indipendenti si ottengono le convoluzioni tra le densità di probabiltià. Nel caso di uniformi tra ± 0.5 si ottiene un triangolo tra ± 1. Queste distribuzioni triangolari possono essere usate per approssimare con una spezzata (in modo lineare) una distribuzione arbitraria.
- Sommando più di due variabili casuali indipendenti si ottengono distribuzioni con andamento polinomiale, che consentono una più accurata rappresentazione di distribuzioni arbitrarie.

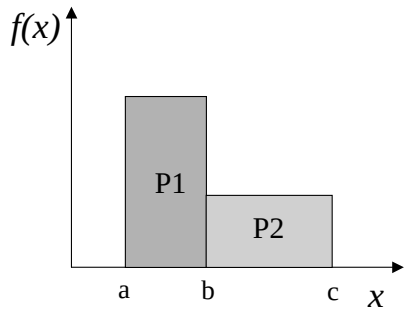


Figura 2.7. Approssimazione di una densità di probabilità con uno scaloide

2.5.4 Generazione della distribuzione gaussiana

La gaussiana non è invertibile analiticamente. Viene di solito generata in uno dei due modi:

- Si sommano 12 variabili casuali uniformi tra ± 0.5 e indipendenti. Per il teorema del limite centrale si tende alla distribuzione gaussiana. La media rimane nulla e la varianza diventa 1 (distribuzione normale). In realtà il supporto è limitato tra ± 6.
- Si approssima l'andamento della gaussiana con archi di sinusoide.

2.6 Test statistici

Per valutare la bontà dei generatori si eseguono test statistici sulle sequenze ottenute. Ci sono tre metodi fondamentali:

- test del χ^2 di Pierson (variabili casuali discrete);
- test di Kolmogorov-Smirnov (variabili casuali continue);
- test di correlazione.

2.6.1 Test di uniformità o del χ^2

Si applica a variabili casuali discrete o discretizzate.

Dati k eventi possibili: E_1,E_2,...,E_k con probabilità p_1,p_2,...,p_k. Si fanno n esperimenti casuali in cui si osservano y_1 eventi di tipo E_1, y_2 eventi di tipo E_2, ..., y_k eventi di tipo E_k, con $\sum_{i=1}^{k} y_i = n$.

La variabile casuale

$$V = \sum_{i=1}^{k} \frac{(y_i - np_i)^2}{np_i}$$

ha una distribuzione detta χ^2 con $k-1$ gradi di libertà. È praticamente indipendente da p_i se n è grande: $np_i > 5\ \forall i$. La variabile V rappresenta una misura della distanza tra le due curve (si veda la figura 2.8).

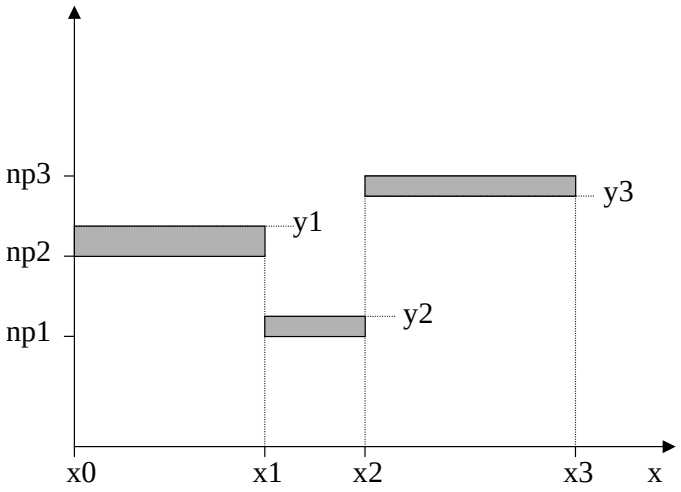


Figura 2.8. Test di uniformità o del χ^2

2.6.2 Test seriale

Si generano dei numeri e si applica il test del χ^2 sulle coppie adiacenti.

2.6.3 Test del Poker

Si considerano come eventi le configurazioni possibili del gioco del poker: coppia, doppia coppia, tris, full e poker. Si applica il test del χ^2 sui cinque numeri successivi.

2.6.4 Test di Kolmogorov-Smirnov

È un metodo che si applica a variabili casuali continue. Fissato un valore di x, si calcola il numero m_x di istanze minori di x all'interno della sequenza:

$$m_x = |\{x_i \mid x_i \leq x\}|$$

e si costruisce una funzione:

$$F_n(x) = \frac{m_x}{n}$$

che si confronta con $F(x)$ (si veda la figura 2.9).

Si ricavano poi i seguenti valori:

$$\begin{aligned} F_n^+ &= \max_n \{F_n(x) - F(x)\} \sqrt{n} \qquad \text{se } F_n(x) \geq F(x) \\ F_n^- &= \max_n \{F(x) - F_n(x)\} \sqrt{n} \qquad \text{se } F_n(x) < F(x) \end{aligned}$$

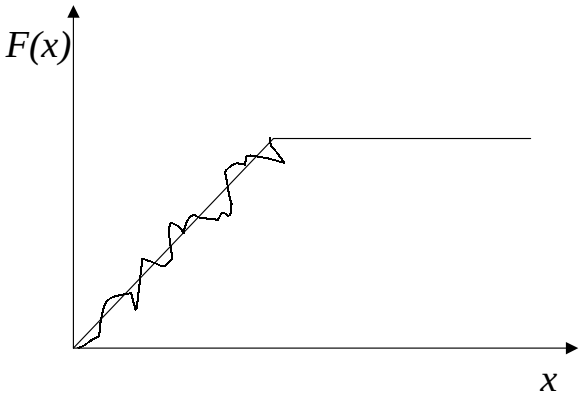


Figura 2.9. Test di Kolmogorov-Smirnov

2.6.5 Test sulla correlazione

Si misura la correlazione tra istanze della variabile casuale:

$$\rho(k) = \frac{\sum_{i=1}^{n} y_i y_{i+k} - \left(\sum_{i=1}^{n} y_i\right)^2 / n}{\sum_{i=1}^{n} y_i^2 - \left(\sum_{i=1}^{n} y_i\right)^2}$$

Deve essere:

$$\frac{1}{\sqrt{n}} < \rho(k) < 0.1$$

2.7 Intervallo e livello di confidenza

Una volta effettuata la simulazione, si deve stimare la precisione e l'affidabilità dei risultati ottenuti.

Dalla simulazione si ricavano n osservazioni:

$$x_1, x_2, \ldots, x_n$$

della variabile casuale x che ha media μ e varianza σ^2. Si calcola la media aritmetica delle osservazioni come:

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

$\overline{x}$ a sua volta è una variabile casuale: ripetendo più volte l'esperimento di simulazione, assume valori diversi, che possono essere caratterizzati come istanze di una vaiabile casuale.

In generale $\overline{x} \neq \mu$, quindi si stima un *intervallo* attorno al valore misurato $\overline{x}$ in modo da prevedere con una certa *confidenza* che μ cada in questo intervallo. Si può facilmente verificare che $\overline{x}$ è uno stimatore imparziale della media, infatti:

$$E[\overline{x}] = \frac{1}{n} \sum_{i=1}^{n} E[x_i] = E[x] = \mu$$

Per la varianza vale:

$$\begin{aligned} var[\overline{x}] &= E\left[(\overline{x} - E[\overline{x}])^2\right] \\ &= E\left[\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)\right)^2\right] \\ &= E\left[\frac{1}{n^2}\left(\sum_{i=1}^{n}(x_i-\mu)^2 + \sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n}(x_i-\mu)(x_j-\mu)\right)\right] \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\sigma^2 = \frac{1}{n}\sigma^2 \end{aligned}$$

Si noti che la varianza di $\overline{x}$ descresce con n, quindi l'incertezza della stima decresce con la durata dell'esperimento di simulazione.

Uno stimatore imparziale della varianza σ^2 è:

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

Si noti come $E\left[S^2\right] = \sigma^2$; quindi S^2 è uno stimatore non polarizzato della varianza.

Se si suppone di sommare un numero molto grande di x_i, per il teorema del limite centrale, lo stimatore $\overline{x}$ ha una distribuzione normale. Effettuando il seguente cambiamento di variabile:

$$z = \frac{\overline{x} - \mu}{\sqrt{\sigma^2}/n}$$

la variabile casuale z ha anch'essa una distribuzione gaussiana con media 0 e varianza 1 (distribuzione normale) essendo $\overline{x}$ una gaussiana (per $n \to \infty$) con media μ e varianza σ^2/n .

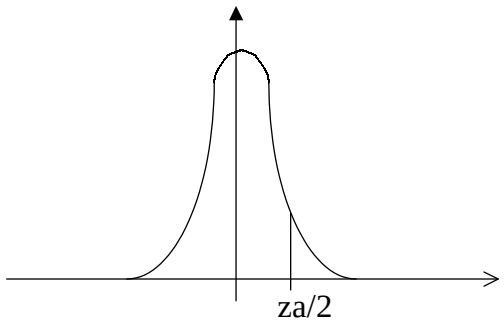


Figura 2.10. Livello e intervallo di confidenza

Considerando la figura 2.10, si ha:

$$
\begin{aligned}
P\left\{-z_{\alpha/2} \leq z \leq z_{\alpha/2}\right\} &= 1-\alpha \\
P\left\{-z_{\alpha/2} \leq \frac{(\overline{x}-\mu)\sqrt{n}}{\sigma} \leq z_{\alpha/2}\right\} &= 1-\alpha \\
P\left\{-z_{\alpha/2}\frac{\sigma}{\sqrt{n}} - \overline{x} \leq -\mu \leq z_{\alpha/2}\frac{\sigma}{\sqrt{n}} - \overline{x}\right\} &= 1-\alpha \\
P\left\{\overline{x} - z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leq \mu \leq \overline{x} + z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right\} &= 1-\alpha
\end{aligned}
$$

L'intervallo casuale $\overline{x} - z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$ è detto **intervallo di confidenza** e $1-\alpha$ è il **livello di confidenza**.

Si puntualizza che "intervallo di confidenza dell'1% con livello di confidenza del 99%", significa che "il 99% delle volte il valore vero μ della media cade in un intervallo di $\pm 1\%$ attorno al valore stimato $\overline{x}$ (e viceversa)".

Siccome σ non è nota, si può usare il suo stimatore S. Si usa invece di z:

$$t = \frac{(\overline{x}-\mu)\sqrt{n}}{S}$$

che è distribuita come una t-student con $n-1$ gradi di libertà.

$$P\left\{\overline{x} - t_{\alpha/2,n-1}\frac{S}{\sqrt{n}} \leq \mu \leq \overline{x} + t_{\alpha/2,n-1}\frac{S}{\sqrt{n}}\right\} = 1-\alpha$$

Per n sufficientemente grande (alcune decine) la t-student si confonde con la normale.

2.8 Metodi di stima del transitorio

Di solito interessa calcolare gli indici di prestazione di un sistema nelle condizioni di funzionamento di regime o stazionarie. Partendo da uno stato qualsiasi il sistema tende a raggiungere il funzionamento di regime dopo un certo periodo, detto di *transitorio*.

Per una certa stima degli indici di prestazione si deve riuscire a stimare la durata del periodo di transitorio per poter poi effettuare le misure degli indici solo a regime.

Il problema legato alla sua stima è dovuto ai seguenti punti:

- è difficile identificare un "buon" stato iniziale;
- è difficile distinguere il funzionamento di regime da quello transitorio;
- esistono solo metodi euristici, tipicamente basati sull'idea che la variabilità nei periodi di transitorio è più elevata che a regime.

I due metodi euristici utilizzati per stimare la lunghezza del transitorio sono:

- **metodo del troncamento**: dati n campioni $\{x_1,x_2,\ldots,x_n\}$ si confronta per ogni $l = 1,2,\ldots,n$ il valore di x_{l+1} con il massimo e il minimo dei campioni nel sottoinsieme $\{x_{l+1},x_{l+2},\ldots,x_n\}$. Il più piccolo l per cui x_{l+1} non è né il minimo né il massimo di $\{x_{l+1},x_{l+2},\ldots,x_n\}$ è la lunghezza del transitorio (si veda la figura 2.11).

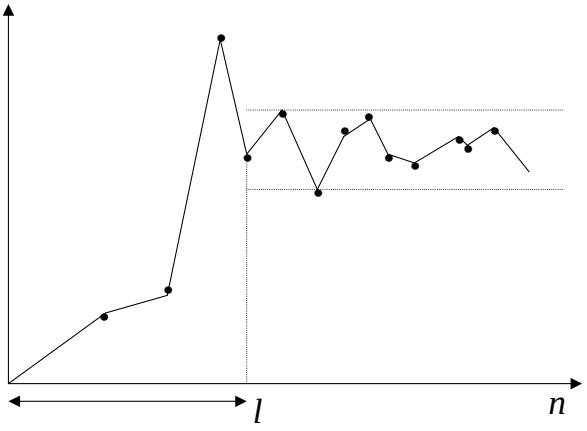


Figura 2.11. Stima del transitorio: metodo del troncamento

- **metodo della cancellazione dei dati iniziali**: dati n campioni $\{x_1,x_2,\ldots,x_n\}$ se ne calcola la media $\overline{x}$. Al crescere di $l = 1,2,\ldots,n$ si calcola la media $\overline{x_l}$ di campioni $\{x_{l+1},x_{l+2},\ldots,x_n\}$ e si ricava la variazione relativa:

$$R_l = \frac{\overline{x_l} - \overline{x}}{\overline{x}}$$

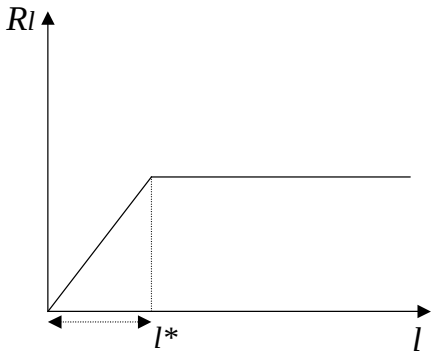


Figura 2.12. Stima del transitorio: metodo della cancellazione dei dati iniziali

Il valore di l^* in corrispondenza del ginocchio della curva identifica la lunghezza del transitorio (si veda la figura 2.12).

2.9 Metodi per avere campioni scorrelati

La teoria degli intervalli di confidenza si basa sull'ipotesi di scorrelazione dei campioni. Esistono tre metodi per generare campioni scorrelati:

- **metodo delle prove ripetute (repeated trials):** i campioni scorrelati sono ottenuti effettuando simulazioni indipendenti, tipicamente ottenute con semi diversi del generatore di numeri casuali. Per ogni simulazione bisogna eliminare il transitorio iniziale. I campioni ottenuti sono facilmente scorrelabili ma risentono delle difficoltà di stima del transitorio iniziale.
- **Metodo Batch means**: si effettua una simulazione molto lunga e la si divide in intervalli (detti batch) indipendenti, la cui lunghezza è un pó maggiore della lunghezza del transitorio. Da questi batch si calcolano campioni secondari "quasi" scorrelati (permane una correlazione tra i bordi dei batch) $y_i = \frac{1}{n_i} \sum_{j=1}^{n} x_j$. Si elimina un solo transitorio iniziale.
- **Metodo rigenerativo:** si identificano alcuni stati del sistema nei quali il comportamento futuro non dipende dalla storia passata (per esempio gli istanti di fine servizio in una coda M/G/1). Gli istanti in cui il sistema entra in questi stati sono detti di rigenerazione: il sistema si rigenera perché perde la correlazione con la storia passata. I campioni ricavati durante intervalli tra due punti di rigenerazione sono scorrelati. Questo metodo non richiede l'eliminazione del transitorio, ma in generale è difficile trovare i punti di rigenerazione e molti sistemi non sono rigenerativi.

3

Protocolli ed architetture di rete

Lo scopo di un protocollo di comunicazione è permettere a due interlocutori di scambiare informazioni utilizzando una rete di telecomunicazioni. Un protocollo di comunicazione è definito da un insieme di regole, che possiamo suddividere in tre categorie:

- *algoritmi*, che specificano come gli interlocutori si comportano per accedere alla rete e come funziona la rete internamente;
- *temporizzazioni*, che determinano le tempistiche di esecuzione degli algoritmi;
- *formati*, che descrivono come devono essere strutturate le informazioni che gli interlocutori si scambiano.

Questa definizione può sembrare molto astratta; in realtà molte azioni quotidiane si basano su di un uso implicito di protocolli di comunicazione. Due esempi vicini alla nostra esperienza sono gli accessi al sistema *telefonico* ed al sistema *postale*. Quando si esegue una telefonata si devono seguire delle regole (un protocollo) per riuscire a comunicare: il protocollo prevede il sollevamento del microtelefono, la composizione del numero telefonico dell'interlocutore, l'attesa della risposta, e così via. Anche per accedere al sistema postale è necessario seguire un protocollo, che prevede ad esempio l'indicazione dell'indirizzo del destinatario in una posizione opportuna sulla busta (altrimenti difficilmente la lettera sarà recapitata).

I protocolli per le reti di telecomunicazioni sono sovente molto articolati. Essi stabiliscono le modalità secondo le quali un utente può accedere ai servizi di rete e i meccanismi secondo cui tali servizi vengono forniti, quindi le regole secondo le quali gli utenti sono in grado di comunicare. Per la realizzazione dei protocolli di rete, le industrie si sono orientate, fin dalla nascita delle reti di telecomunicazioni, verso *architetture a strati* (si veda la figura 3.1). Tali strutture sono definite suddividendo l'insieme di regole per la comunicazione tra utenti in una serie di "protocolli più semplici" che, combinati, realizzano funzioni via via più complesse. Ogni strato (o livello) fornisce *servizi* agli strati superiori e li realizza basandosi sui servizi ad esso forniti dagli strati inferiori. Ovviamente gli strati superiori operano ad un livello logico di maggior complessità, e forniscono servizi più sofisticati rispetto agli strati inferiori.

Ogni strato è in grado di richiedere o fornire servizi solamente agli strati adiacenti e ignora completamente che cosa accada negli altri strati della architettura; gli strati adiacenti comunicano mediante le *primitive di interfaccia*.

Queste caratteristiche comportano molti vantaggi: permettono di semplificare la fase di progetto, di manutenzione e di aggiornamento del software, poiché il singolo strato è molto più semplice dell'insieme di regole complessivo; permettono di modificare uno strato del protocollo senza dover intervenire sugli altri

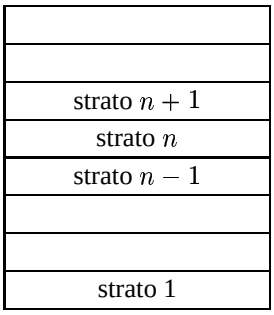


Figura 3.1. Architettura a strati

strati, purché vengano rispettate le interfacce con gli strati adiacenti; inoltre, i servizi forniti dagli strati inferiori possono essere utilizzati da più strati adiacenti superiori contemporaneamente.

Questi concetti sono presenti in tutte le architetture delle reti di comunicazioni sviluppate verso la fine degli anni '60, quali ad esempio ARPANET, SNA (IBM) e DNA (Digital). Purtroppo però la stratificazione non è avvenuta nello stesso modo nelle diverse architetture: tutte le architetture di rete citate sono a strati, ma gli strati sono diversi tra loro, quindi le architetture non sono compatibili.

Questa situazione di incompatibilità ha portato alla necessità di stabilire standard comuni; l'ISO è l'organismo internazionale che si è occupato della standardizzazione delle architetture di rete e ha proposto una famiglia di standard nota con il nome OSI (Open Systems Interconnection), che ne descrive le caratteristiche.

3.1 Il Modello OSI

Il modello di riferimento OSI [3.6] definisce l'architettura di *sistemi*, detti *sistemi aperti*, disposti a scambiare informazioni mediante una rete di telecomunicazioni eterogenea. La struttura di una rete formata da quattro sistemi è riportata nella figura 3.2.

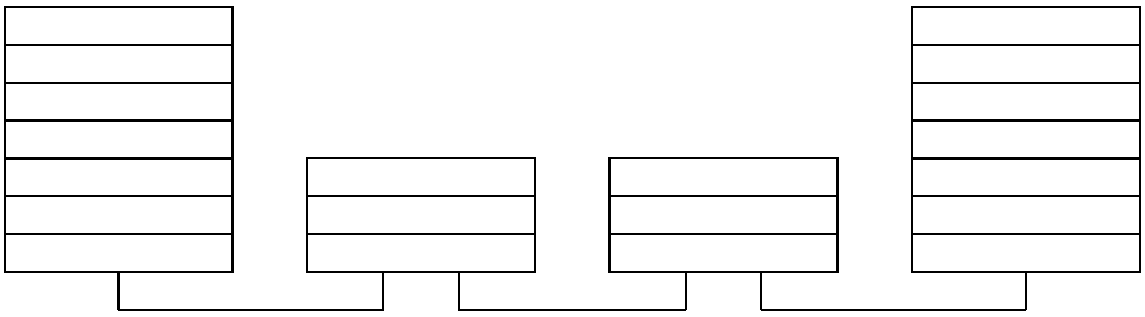


Figura 3.2. Modello OSI di una rete con quattro sistemi

Ogni sistema utente è costituito da sette strati o *livelli*. I sistemi intermedi (detti sistemi "relay") invece comprendono solo i tre livelli più bassi. L'insieme dei sette (o tre) strati viene anche chiamato pila o piano (di utente, in contrapposizione al piano di gestione, di cui parleremo brevemente nel seguito). In pratica, le pile di 7 livelli corrispondono ai sistemi di utente, mentre le pile di 3 livelli corrispondono ai nodi di commutazione della rete. Il livello superiore dei sistemi utente (7) si interfaccia direttamente con l'utente, mentre il livello inferiore (1) si interfaccia con il mezzo trasmissivo. In ogni livello sono presenti una o più *entità* che sono gli interlocutori della comunicazione; le entità presenti nel livello N sono dette (N)entità e comunicano con altre (N)entità di altri sistemi mediante un (N)*protocollo*.

Ogni entità è un processo che fornisce *servizi* alle entità di livello superiore mediante *funzioni* che sfruttano i servizi offerti dall'entità di livello inferiore, ai quali accede attivando delle *primitive di interfaccia*.

Non si confondano servizi, funzioni e primitive: le funzioni sono operazioni svolte all'interno di un determinato livello, i servizi sono offerti su un'interfaccia tra livelli adiacenti, mentre le primitive permettono di attivare i servizi.

Le entità sfruttano i servizi per aprire *connessioni* con altre entità dello stesso livello; le connessioni permettono lo scambio di informazioni tra entità dello stesso livello.

Supponiamo che una (N)entità del sistema A voglia aprire una comunicazione con una (N)entità del sistema B (si veda la figura 3.3). La (N)entità chiede un (N−1)servizio al livello (N−1); per realizzare questo servizio è necessario far avvenire uno scambio di informazioni tra tutti i livelli sottostanti al livello N del sistema A. Il flusso di informazioni segue quindi in ogni sistema un percorso "verticale". L'informazione arriva al livello inferiore e viene trasferita sul canale. Sul sistema B l'informazione seguirà il percorso inverso, risalendo attraverso tutti gli strati fino ad arrivare al livello N.

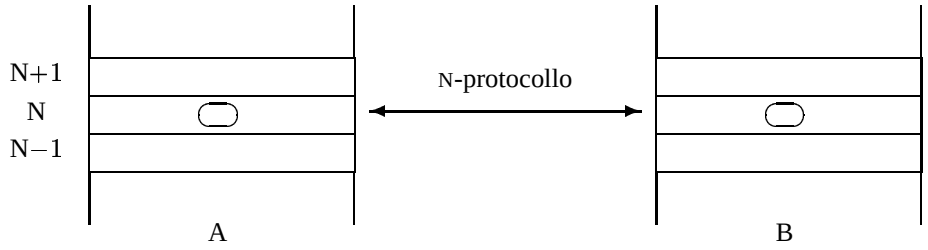


Figura 3.3. (N) entità nei sistemi A e B e protocollo di livello N

Ognuna delle due (N)entità vede tutto il sistema come una "scatola nera", che risponde come una (N−1)entità e che la mette in comunicazione con un'altra (N)entità (si veda la figura 3.4). L'informazione si sposta fisicamente all'interno di ogni sistema in "verticale", ma le due entità utilizzano un protocollo di livello N che prescinde dal percorso reale dell'informazione e che simula uno scambio di informazioni in "orizzontale".

Esaminiamo ora in modo dettagliato i vari elementi dell'architettura OSI; in particolare i livelli, le entità, i protocolli e le primitive.

3.2 Livelli

L'architettura OSI è organizzata in sette livelli:

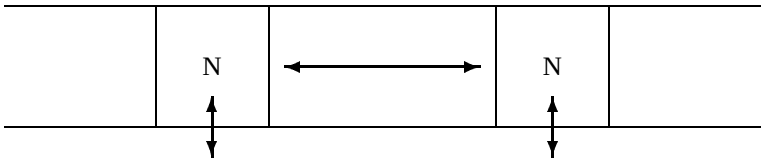


Figura 3.4. Protocollo di livello N

- livello 1 o fisico;
- livello 2 o collegamento;
- livello 3 o rete;
- livello 4 o trasporto;
- livello 5 o sessione;
- livello 6 o presentazione;
- livello 7 o applicazione.

Prima di descrivere brevemente i diversi livelli, si osservi che, poiché l'architettura OSI è molto generale, l'importanza di ogni livello è molto diversa a seconda del tipo di rete che si considera: alcuni livelli assumono un'importanza fondamentale in una rete locale che usa un canale broadcast, ma sono trascurabili o inesistenti in una rete pubblica con topologia a maglia.

Il livello **fisico** (*physical*) è il livello di interfaccia con il canale e descrive tutti gli aspetti di carattere trasmissivo. Il livello fisico definisce ad esempio gli standard di modulazione, i codici di linea, i livelli di tensione, la temporizzazione dei bit, il recupero del sincronismo di bit, la mecccanica dei connettori per il collegamento con la linea fisica, e così via. L'unità di informazione elementare scambiata a livello fisico è il *bit*.

Il livello **collegamento** (*data link*) è diviso in due sottolivelli: il sottolivello 2.1 detto MAC (*Medium Acces Control*) e il sottolivello 2.2 detto LLC (*Logical Link Control*). Il sottolivello MAC definisce gli standard per la condivisione di un canale comune, mentre il sottolivello LLC cerca di garantire un trasferimento tra sistemi direttamente collegati da un canale di comunicazione di sequenze di bit "esenti" da errore e gestisce il controllo di flusso e la segmentazione delle sequenze di bit. L'unità elementare è la *stringa* di bit, detta anche *trama*.

Il livello **rete** (*network*) gestisce la rete come insieme di canali di comunicazione e di nodi. Le problematiche fondamentali del livello rete sono l'instradamento, la tariffazione e il controllo di congestione. L'unità di informazione elementare a questo livello è il *pacchetto*.

Questi primi tre livelli formano quella che viene di solito denominata *sottorete di comunicazione*.

Il livello **trasporto** (*transport*) è il primo livello che si occupa della comunicazione diretta tra due utenti (*end-to-end:* dal sistema sorgente al sistema destinazione) e la sua unità di informazione elementare è il *messaggio*. Tra i suoi compiti principali si ricordano la sequenzializzazione dei messaggi, il controllo di errore e di flusso tra utente e utente e la frammentazione dei messaggi in pacchetti all'utente sorgente ed il riassemblaggio dei messaggi all'utente destinatario.

Il livello **sessione** (*session*) gestisce il colloquio tra due utenti. Le sue funzioni principali sono la strutturazione e la sincronizzazione del dialogo.

Il livello **presentazione** (*presentation*) si occupa del formato di rappresentazione dei dati. Tra le funzioni principali si ricordano la crittografia e la compressione dell'informazione trasmessa.

Il livello **applicazione** (*application*) implementa servizi di informatica distribuita. Si interfaccia direttamente con l'utente; tra i suoi servizi si ricordano il trasferimento di file, l'emulazione di terminale, la posta elettronica, e l'accesso a banche dati.

3.3 Entità, funzioni di indirizzamento e connessioni

Le *entità* sono gli interlocutori dello scambio di informazioni tra due sistemi. Una (N)entità, per scambiare informazioni con una entità di pari livello (detta *peer entity*), attiva primitive di servizio fornite dal livello $N-1$; deve però fornire l'indirizzo della (N)entità con cui desidera aprire una connessione. L'indirizzo può essere ottenuto mediante le *funzioni di indirizzamento*.

Ogni entità è univocamente identificata da un *titolo*; il titolo è un identificatore assoluto, che permette l'identificazione delle entità in tutta la rete e non dipende dalla posizione fisica dell'entità all'interno della rete. Infatti le entità, essendo processi in un ambiente distribuito, possono spostarsi da un sistema all'altro. Poiché il titolo non dipende dalla posizione dell'entità, sono necessarie funzioni di trasformazione del titolo in un indirizzo relativo, legato alla posizione momentanea dell'entità che si vuole raggiungere.

Ogni (N)entità è collegata all'interfaccia con il livello $N-1$ per mezzo di un $(N-1)$SAP (*Service Access Point*). Gli (N)SAP sono identificati mediante (N)indirizzi fissi, legati all'interfaccia tra livello e livello in un determinato sistema. Quando si stabilisce una connessione tra due (N)entità, queste vengono identificate per mezzo degli $(N-1)$indirizzi degli $(N-1)$SAP ai quali esse sono collegate.

Esiste una funzione di livello N detta (N)*direttorio* che stabilisce una corrispondenza tra gli (N)titoli e gli $(N-1)$indirizzi degli $(N-1)$SAP attraverso cui le (N)entità accedono ai $(N-1)$servizi Poiché una (N)entità è indirizzata tramite l'$(N-1)$SAP a cui è collegata, ovvero mediante un $(N-1)$indirizzo, la funzione di (N)direttorio permette ad una (N)entità di conoscere l'indirizzo della (N)entità remota con cui vuole comunicare.

In ogni (N)SAP esistono degli (N)CEP (*Connection End Point*), che permettono di aprire più connessioni utilizzando lo stesso (N)SAP. Gli (N)CEP rappresentano gli "estremi" del circuito virtuale che collega le due $(N+1)$entità che hanno stabilito la connessione. Essi vengono identificati tramite identificatori con significato locale detti (N)CEI (*Connection End Point Identifier*).

Supponiamo per esempio che la $(N+1)$entità della figura 3.5 voglia aprire una connessione con una $(N+1)$entità di un altro sistema. Conosce l'$(N+1)$titolo dell'entità destinataria; accede all'$(N+1)$direttorio per ottenere l'(N)SAP a cui l'entità destinataria è collegata. Questa operazione si ripete per ogni livello fino al livello fisico e, sul sistema destinatario, dal livello fisico fino al livello N, in modo da creare un collegamento virtuale tra le due entità.

Inoltre, poiché una entità è collegata mediante SAP sia al livello inferiore che al livello superiore, è prevista all'interno del livello N una funzione di (N)*mapping* che definisce la corrispondenza tra gli (N)SAP attraverso i quali le (N)entità forniscono (N)servizi e gli $(N-1)$SAP utilizzati per accedere agli $(N-1)$servizi. Questa corrispondenza può essere realizzata in tre modi diversi, come riportato nella figura 3.6:

- collegamento uno a uno;
- collegamento gerarchico;
- collegamento mediante tabelle di conversione.

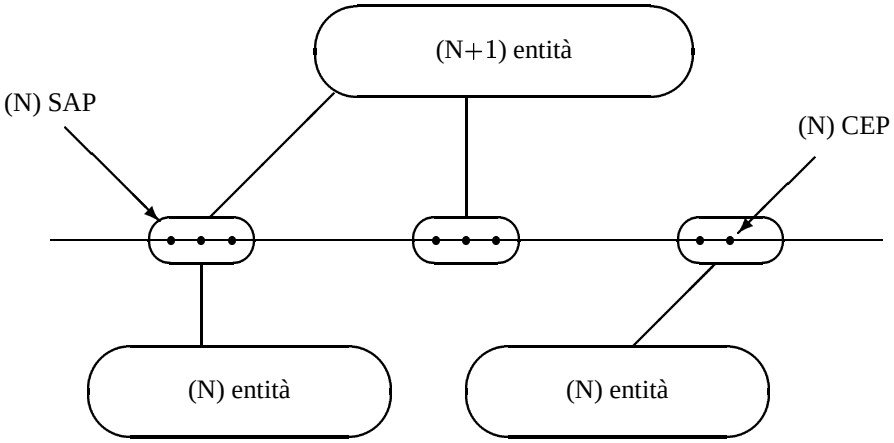


Figura 3.5. Rappresentazione di entità, SAP e CEP

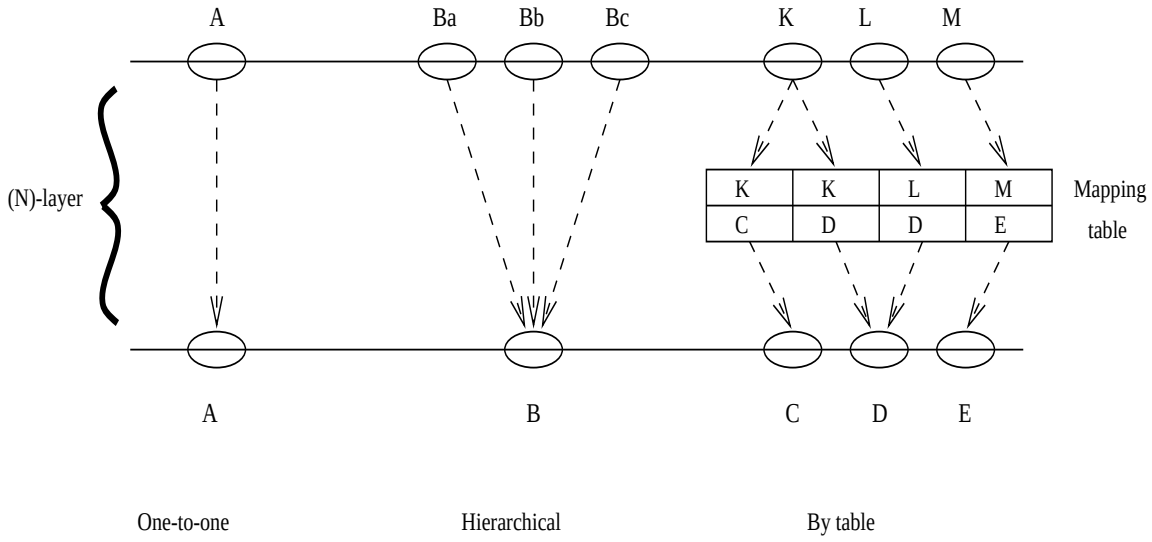


Figura 3.6. Possibilità di collegamento tra entità di livelli adiacenti

L'architettura OSI prevede che lo scambio di informazioni tra due (N+1)entità utilizzi (N)connessioni stabilite tra due (N)SAP. In generale per creare una (N)connessione possono essere usate una o più (N−1)connessioni. La relazione tra (N) e (N−1)connessioni può essere di tre tipi, come riportato nella figura 3.7:

uno a uno: ad una (N)connessione corrisponde una (N−1)connessione;

multiplexing: più (N)connessioni corrispondono ad una (N−1)connessione;

splitting: una (N)connessione è realizzata mediante più (N−1)connessioni.

Lo splitting può essere utile per ottenere prestazioni migliori di quelle fornite da una sola (N−1)connessione. Il multiplexing può essere utile per risparmiare risorse di rete, nel caso in cui si possano far coesisitere con prestazioni adeguate più (N) connessioni su una (N−1)connessione.

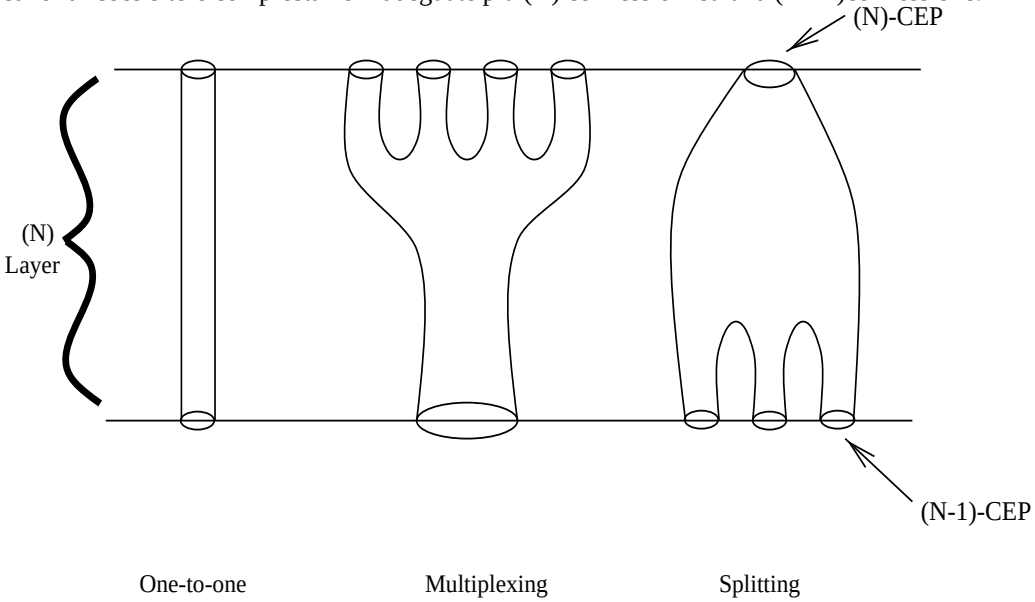


Figura 3.7. Connessioni nel sistema ISO-OSI

3.4 Protocolli e formati delle unità dati

L'architettura OSI si riferisce a reti di telecomunicazioni a commutazione di pacchetto; quando una (N+1)entità desidera trasmettere un pacchetto ad una (N+1)entità remota, accede al (N)servizio e chiede la trasmissione di una (N)SDU (*Service Data Unit* di livello N) al destinatario.

La (N)SDU passando attraverso l'interfaccia tra il livello N+1 ed il livello N può venire convertita in una (N)IDU (*Interface Data Unit*). Tale IDU è utilizzata esclusivamente laddove è necessaria una conversione di formato affinché la (N)entità interpreti correttamente i dati provenienti dalla (N+1)entità ad essa collegata (e viceversa). L'entità di livello N crea poi una (N)PDU (*Protocol Data Unit*) aggiungendo alla (N)SDU una (N)PCI (*Protocol Control Information)* che contiene le informazioni necessarie al protocollo di questo livello. Nella figura 3.8 è riportato il processo di scambio di informazioni tra due livelli adiacenti.

È evidente che una (7)PDU (che contiene fra gli altri i dati originariamente prodotti dall'utente), nel processo di trasformazione che subisce scendendo ai livelli inferiori, produce, in assenza di operazioni di segmentazione, PDU di dimensioni sempre maggiori, poiché ogni livello aggiunge la PCI necessaria per il proprio protocollo (si veda la figura 3.9).

In realtà il processo di passaggio da un livello a quello inferiore non è semplice come descritto: una (N)PDU, nel diventare una (N−1)PDU può subire un processo di *segmentazione*, cioè di scomposizione in

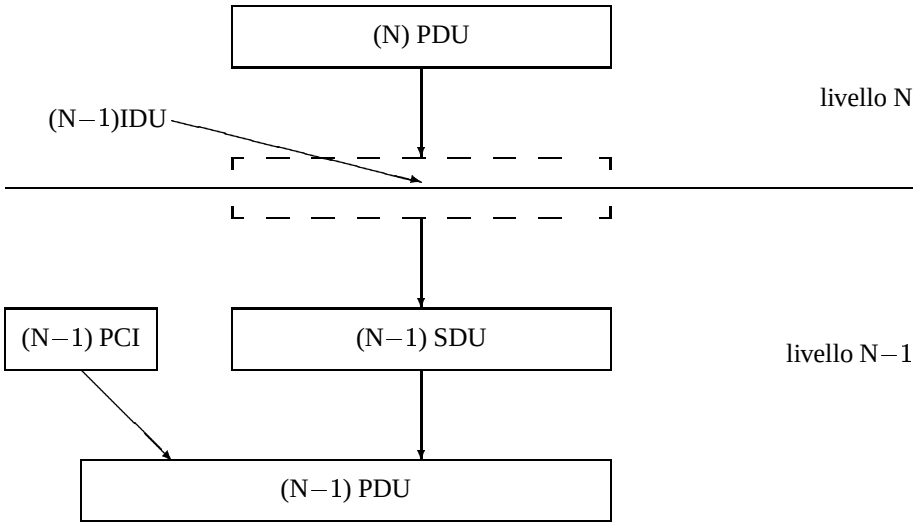


Figura 3.8. Passaggio di una PDU da livello N a livello N−1

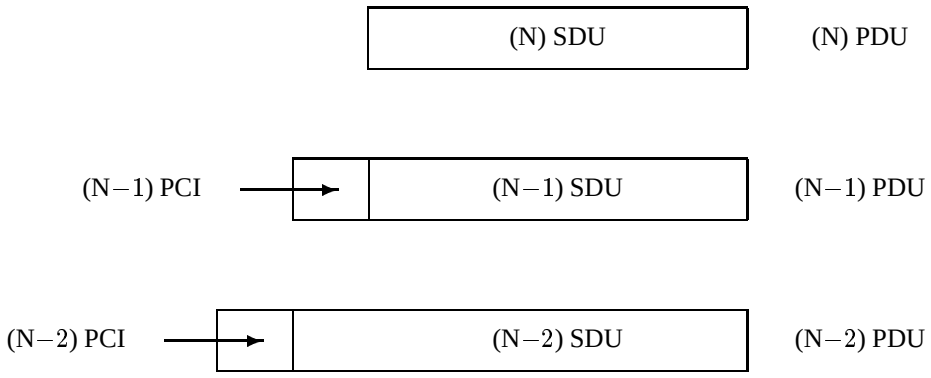


Figura 3.9. Processo di incapsulamento delle PDU

più PDU di dimensioni inferiori; più raramente alcune (N)PDU possono essere raggruppate per formare una unica (N−1)PDU in un processo detto di *blocco*.

3.5 Primitive

Le *primitive* sono procedure che permettono di attivare i servizi forniti dal livello inferiore.

In ambito OSI sono previsti quattro tipi di primitive:

- richiesta (*request*)
- indicazione (*indication*)
- risposta (*response*)
- conferma (*confirm*)

Qualunque primitiva OSI rientra in una di queste categorie.

Per descrivere le primitive si utilizzano *diagrammi temporali* simili a quello riportato nella figura 3.10, in cui l'asse dei tempi cresce verso il basso, mentre orizzontalmente corre una coordinata spaziale. La zona centrale rappresenta la rete e tutti i protocolli dal livello (N−1) fino al livello 1 sui due sistemi, mentre a sinistra e a destra sono rappresentate le due (N)entità che vogliono scambiarsi informazioni.

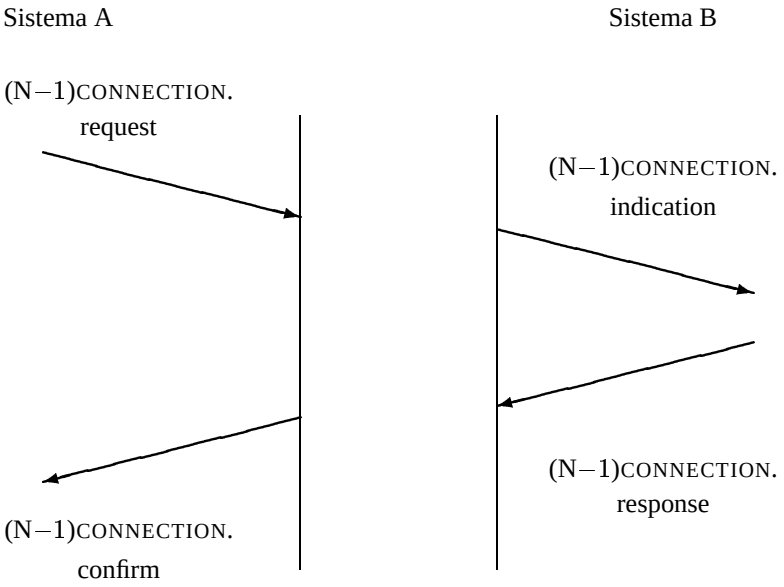


Figura 3.10. Fasi di apertura di connessione

Ad esempio la richiesta di apertura di una connessione viene eseguita nelle fasi seguenti.

- La (N)entità del sistema chiamante A attiva la primitiva (N−1)CONNECTION.request.
- L'(N−1)entità di A, attraverso una serie di fasi intermedie, trasmette l'informazione relativa alla richiesta di apertura di una connessione alla (N−1)entità del sistema ricevente B, che attiverà la primitiva (N−1)CONNECTION.indication.
- La (N)entità del sistema B, se accetta la richiesta di connessione, attiva la primitiva (N−1)-CONNECTION.response.

- La (N−1)entità del sistema B, attraverso una serie di fasi intermedie, trasmette l'informazione relativa alla accettazione della apertura di una connessione alla (N−1)entità del sistema A, che attiva la primitiva (N−1)CONNECTION.confirm che informa l'(N)entità dell'avvenuta connessione.

Da questo momento in poi è aperta una connessione tra le (N)entità del sistema A e del sistema B.

Esistono primitive OSI che permettono di rifiutare la connessione in questa fase iniziale o di chiuderla successivamente e primitive per lo scambio di dati. Le primitive saranno descritte in modo approfondito quando si studieranno i diversi livelli del modello di riferimento OSI, ma il susseguirsi delle primitive è tipicamente analogo a quello appena visto.

3.6 Problemi di gestione della rete

Il principale obiettivo delle funzioni di gestione della rete è quello di garantire la *qualità del servizio* (QOS – *Quality Of Service*) da fornire all'utente.

Per gestire una rete è necessario scambiare informazioni di gestione tra sistemi diversi, però lo scambio di informazioni non avviene come nei sette livelli del piano utente. Le funzioni di gestione di rete sono realizzate in un piano di gestione, parallelo al piano utente, in cui non esiste la suddivisione in livelli.

Funzioni tipiche della gestione di rete sono la configurazione, il monitoraggio, la diagnostica e la riconfigurazione in seguito a situazioni di congestione o di malfunzionamento della rete.

Riferimenti bibliografici

ISO International Standard 7498, *Data Processing – Open Systems Interconnection – Basic Reference Model*

4

Protocolli a finestra

Prima di descrivere in qualche dettaglio i protocolli di alcuni livelli dell'architettura OSI, è utile descrivere un algoritmo generale, utilizzato in numerosi livelli per il trasferimento di unità di informazione su di una connessione, che viene normalmente chiamato *protocollo a finestra*.

I tre problemi principali che deve risolvere un protocollo di comunicazione sono:

- il controllo di errore: il trasferimento dati deve avvenire con il minor numero di errori possibile;
- il controllo di flusso: la velocità di trasferimento dei dati verso il ricevitore deve essere inferiore alla capacità del ricevitore di accettare ed elaborare i dati;
- il mantenimento (o la ricostruzione) dell'ordine tra le unità date trasferite: il ricevitore deve essere in grado di ricostruire la sequenza delle unità dati trasferite.

I protocolli a finestra utilizzano due tipi di PDU: una permette di trasferire l'informazione utile sul canale, l'altra contiene la conferma dell'avvenuta corretta ricezione. La PDU di conferma è detta ACK (dall'inglese *ACKnowledgment*), mentre la PDU che contiene l'informazione sarà indicata con la sigla DT (dall'inglese *DaTa*).

Questo meccanismo è molto simile alla ricevuta di ritorno (o avviso di ricevimento) del sistema postale. Il trasmettitore (mittente) è sicuro dell'avvenuta ricezione del dato (lettera) da parte del ricevitore (destinatario) quando riceve l'ACK (ricevuta di ritorno).

Le PDU vengono numerate dal trasmettitore in modo da permettere il mantenimento della sequenza. Questo significa che le PDU hanno nella loro intestazione (PCI) un campo riservato alla numerazione. Tale campo rappresenta ovviamente uno spreco (*overhead*) introdotto dal protocollo a finestra, in quanto si dedica parte della risorsa trasmissiva al trasferimento di informazione di controllo; esso è composto da un numero finito di bit, il che implica che i numeri di sequenza vengono estratti da un insieme finito, per cui si ripetono ciclicamente.

4.1 Protocollo *stop-and-wait*

Esistono diverse varianti del protocollo a finestra. Il primo che studiamo è il protocollo *stop-and-wait*, detto anche protocollo *alternating bit*. Come vedremo meglio più avanti, è un protocollo a finestra in cui le finestre del ricevitore e del trasmettitore hanno ampiezza unitaria.

Secondo questo algoritmo, il trasmettitore invia una PDU di tipo DT al ricevitore, si ferma (*stop*) e si mette in attesa (*wait*) della PDU di tipo ACK. Solo dopo aver ricevuto l'ACK dal ricevitore, il trasmettitore può

inviare la PDU di tipo DT successiva. Se il trasmettitore non riceve l'ACK, dopo un certo ritardo detto *tempo di timeout* (si noti l'aspetto di temporizzazione, che affianca quello algoritmico su cui ora ci concentriamo, e quello relativo ai formati), ripete la trasmissione della PDU di tipo DT.

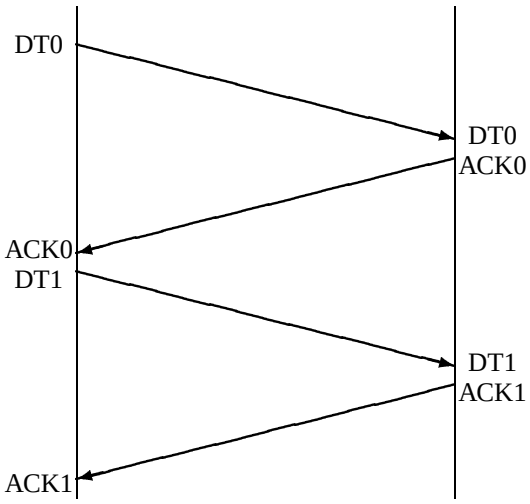


Figura 4.1. Diagramma temporale: nessun errore

Per capire il funzionamento del protocollo *stop-and-wait* utilizziamo i diagrammi temporali. Studiamo inizialmente il caso in cui non si verificano errori (si veda la figura 4.1). Il trasmettitore invia una PDU di tipo DT cui è assegnato il primo numero di sequenza, che supponiamo essere lo zero; tale PDU viene denominata DT0. Dopo un tempo pari al tempo di propagazione, tale PDU arriva al ricevitore, che invia un una PDU di tipo ACK relativa al numero di sequenza 0, che chiameremo ACK0. Il trasmettitore, dopo un tempo pari al tempo di propagazione, riceve l'ACK0 e può trasmettere la PDU di tipo dati successiva, DT1, e così via.

Si ponga attenzione al fatto che è di importanza fondamentale numerare le PDU per evitare malfunzionamenti in caso di errori sul canale. Sul canale si possono perdere PDU di tipo DT (si veda la figura 4.2) e ACK (figura 4.3). Esaminiamo i due casi.

Nel primo caso si perde la PDU DT0. Il ricevitore, non avendo ricevuto nulla, non fa nulla. Il trasmettitore non riceve la PDU ACK0 e, dopo un tempo pari al tempo di timeout, ritrasmette DT0. In questo caso non sembra indispensabile numerare le PDU.

Esaminiamo il secondo caso. La PDU DT0 arriva correttamente al ricevitore, che invia la PDU ACK0; ACK0 si perde, quindi il trasmettitore non riceve nulla. Il trasmettitore non è in grado di discriminare tra questo caso e il caso precedente, poiché non è a conoscenza di ciò che è avvenuto sul canale. Dopo un tempo pari al tempo di timeout invia nuovamente DT0; il ricevitore riceve DT0 e deve potersi accorgere che tale PDU è già stata ricevuta in precedenza. Ciò si ottiene numerando le PDU (si veda la figura 4.4) in modo da sapere riconoscere le ripetizioni (sia per le PDU di tipo DT che per quelle di tipo ACK). Il ricevitore, accortosi che la PDU DT0 è ripetuta, la scarta e invia nuovamente la PDU ACK0. Tale ripetizione di ACK0 è necessaria, affinchè il trasmettitore possa procedere con la trasmissione di DT1.

Esistono ovviamente casi più complicati, come quello riportato nella figura 4.5. In questo caso si ha un ricevitore molto lento rispetto al trasmettitore; la numerazione diviene importante anche per il trasmettitore, che altrimenti non sarebbe in grado di distinguere un ACK0 arrivato in ritardo da un ACK1 relativo alla PDU di tipo DT successiva.

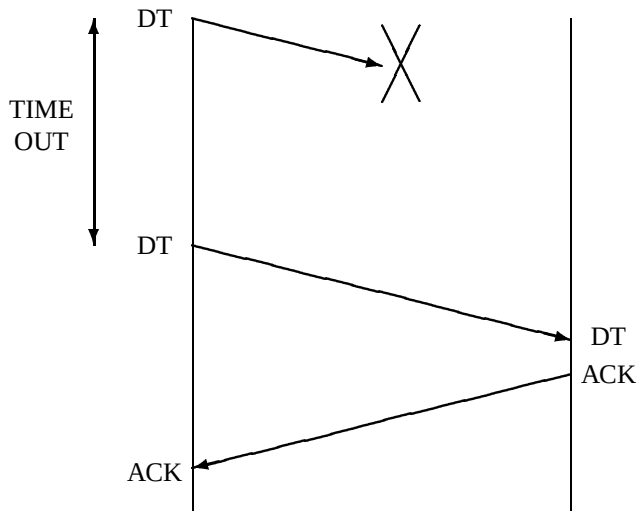


Figura 4.2. Diagramma temporale: perdita di PDU di tipo DT

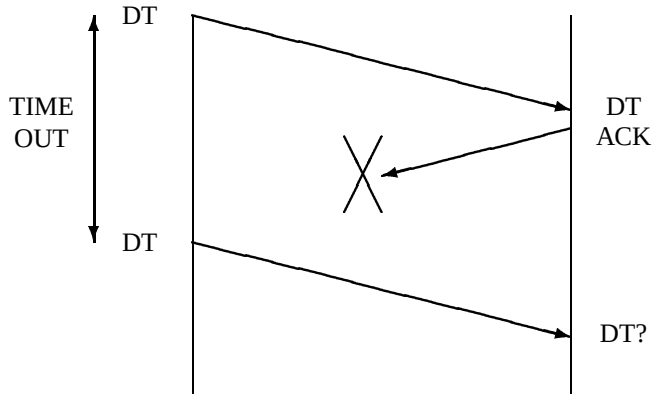


Figura 4.3. Diagramma temporale: perdita di PDU di tipo ACK

Esistono molte altre combinazioni possibili di ritardi e di errori sul canale che potrebbero dare luogo a errori di protocollo, se non si numerassero le unità dati.

Le regole fondamentali del protocollo *stop-and-wait* sono quindi:

- il trasmettitore trasmette una PDU numerata DTn e si mette in attesa;
- il ricevitore trasmette sempre una PDU ACKn dopo aver ricevuto una PDU DTn;
- il trasmettitore, dopo un tempo pari al tempo di timeout, ritrasmette la stessa PDU;
- il trasmettitore trasmette la PDU successiva DT$n+1$ solo se ha ricevuto l'ACKn;

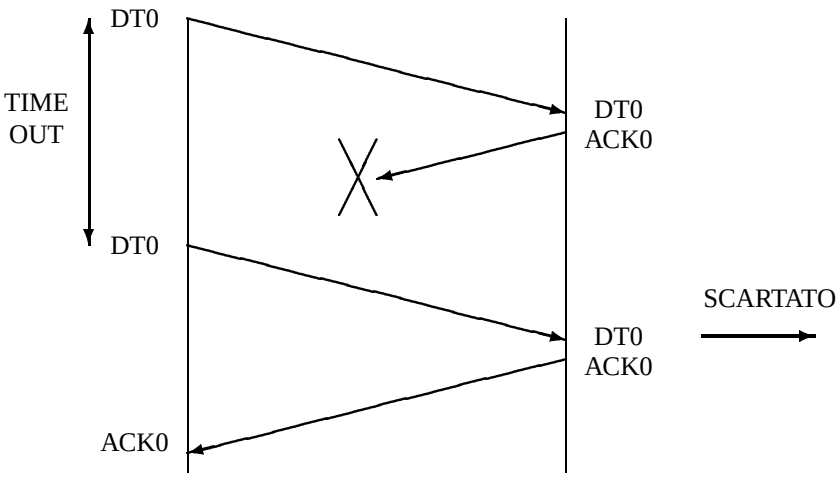


Figura 4.4. Diagramma temporale: numerazione delle PDU

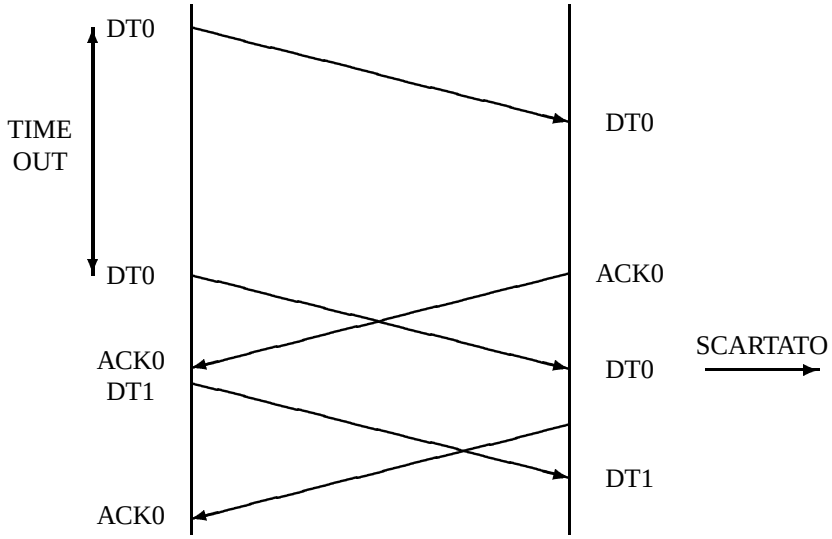


Figura 4.5. Diagramma temporale: ricevitore lento

- tutte le PDU devono essere numerate; per la numerazione è sufficiente un bit, ovvero si devono potere distinguere due PDU consecutive dello stesso tipo.

Il fatto che per la numerazione sia sufficiente un solo bit spiega il nome *alternating bit protocol*, talvolta utilizzato per descrivere questo protocollo.

Se il canale non mantiene la sequenza delle PDU trasmesse, il protocollo non funziona; si possono cioè avere errori se il canale non è sequenziale (non sequenzialità sono normali in reti a commutazione di pacchetto

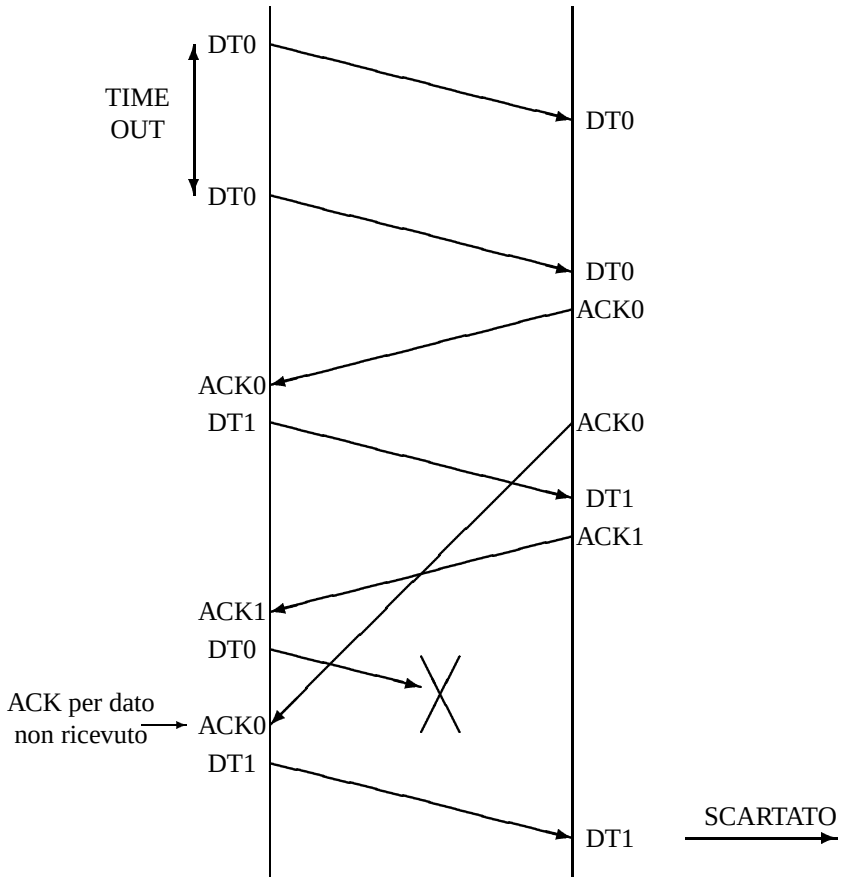


Figura 4.6. Diagramma temporale: canale non sequenziale

con funzionamento di tipo datagram). Questo fenomeno è evidenziato nella figura 4.6, nella quale un ACK che subisce un forte ritardo rispetto alle altre PDU trasferite causa la perdita di due PDU di tipo DT. Si noti come uno schema di numerazione delle PDU con più numeri di sequenza permetta di ridurre l'incidenza di situazioni del tipo mostrato nella figura 4.6. Si veda a questo riguardo la figura 4.7, dove la situazione mostrata nella figura 4.6 viene riproposta nel caso in cui le PDU vengono numerate con due bit. Ovviamente numerare le PDU con un numero maggiore di bit migliora l'affidabilità del protocollo, ma aumenta l'overhead (quindi diminuisce l'efficienza) in quanto sono necessari più bit nella PCI delle PDU.

La descrizione appena vista del protocollo *stop-and-wait* riguarda il caso in cui si ha una trasferimento di dati unidirezionale, cioè da una entità trasmittente ad una ricevente. Quando il flusso dati è bidirezionale, cioè entrambe le entità coinvolte nella comunicazione agiscono sia da trasmettitori sia da ricevitori, è possibile comprendere nella intestazione delle PDU di tipo DT un campo riguardante l'informazione di riscontro per il trasferimento di dati che sta simultaneamente avvenendo nella direzione opposta, limitando così l'invio di PDU di tipo ACK. Questa tecnica è nota in letteratura con il termine *piggybacking*, e viene sovente utilizzata nelle implementazioni dei protocolli a finestra.

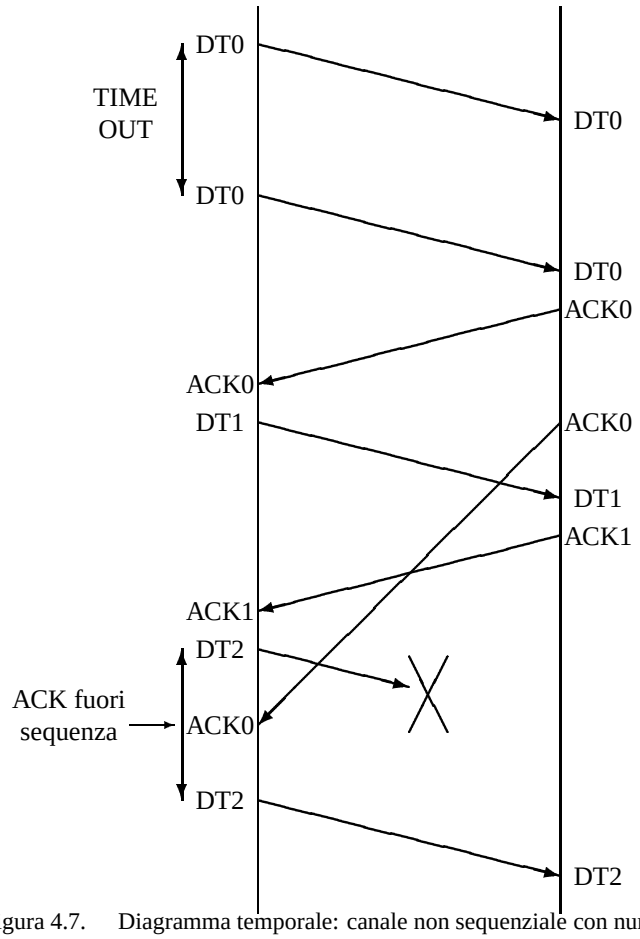


Figura 4.7. Diagramma temporale: canale non sequenziale con numerazione su due bit

4.2 Modello del protocollo *stop-and-wait* con reti di Petri

Modelliamo ora il protocollo *stop-and-wait* usando una rete di Petri. Il modello iniziale descrive il funzionamento del protocollo senza errori ed è riportato nella figura 4.8a. Si aggiunge poi (figura 4.8b) la possibilità di perdita di dati e di ACK. Si tiene conto del meccanismo di timeout (necessario per evitare il fenomeno di *deadlock* (blocco) del protocollo (figura 4.8c). Inoltre è necessario ripetere l'ACK se il dato è duplicato (figura 4.8d). Infine il trasmettitore deve eliminare gli ACK duplicati (figura 4.9a). Gli archi inibitori della figura 4.9b garantiscono il mantenimento della sequenza sul canale.

4.3 Interpretazione del protocollo *stop-and-wait*

Diamo ora una interpretazione del protocollo *stop-and-wait* come protocollo a finestra.

Si immagini di disporre lungo un asse discreto i numeri di sequenza delle PDU di tipo DT che vengono

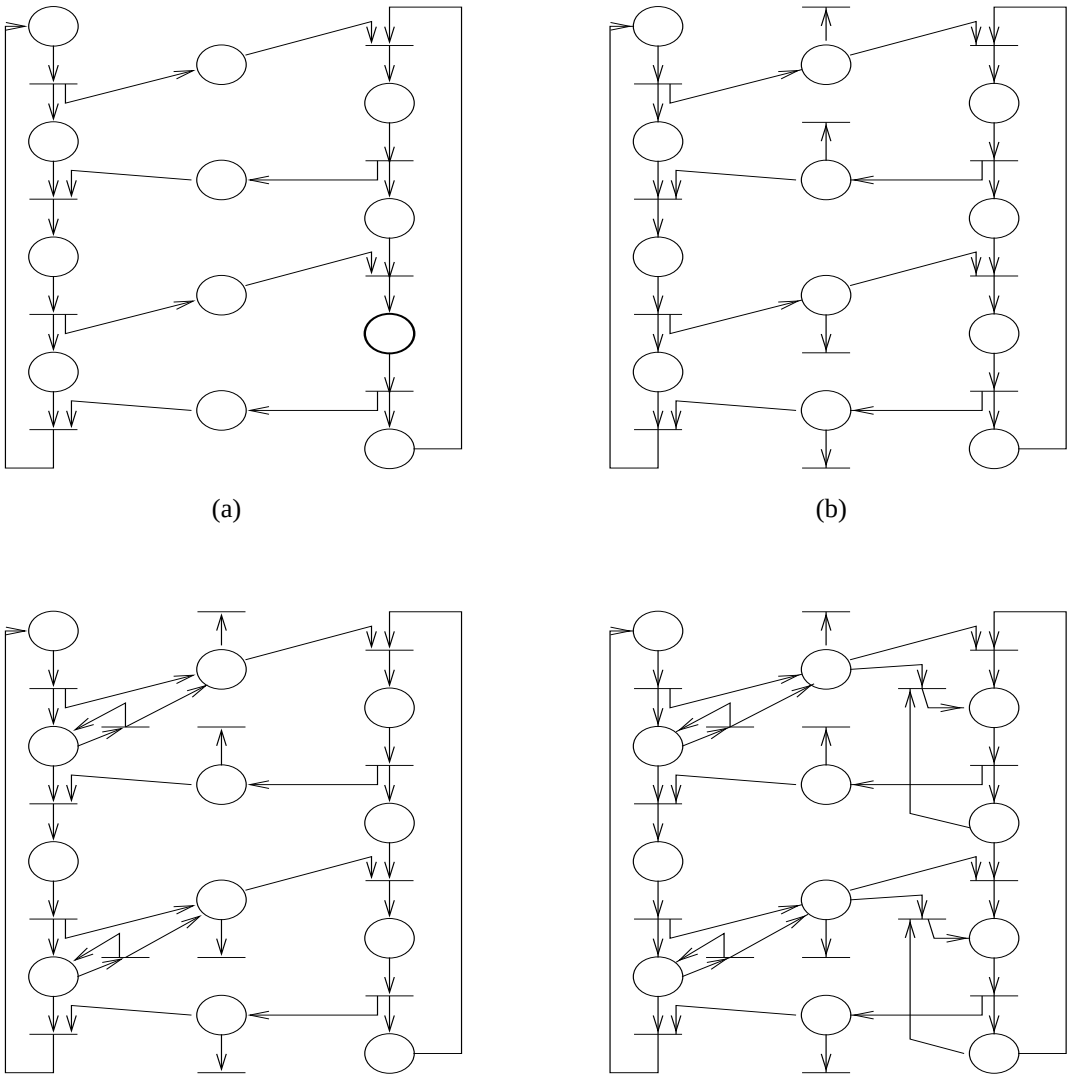
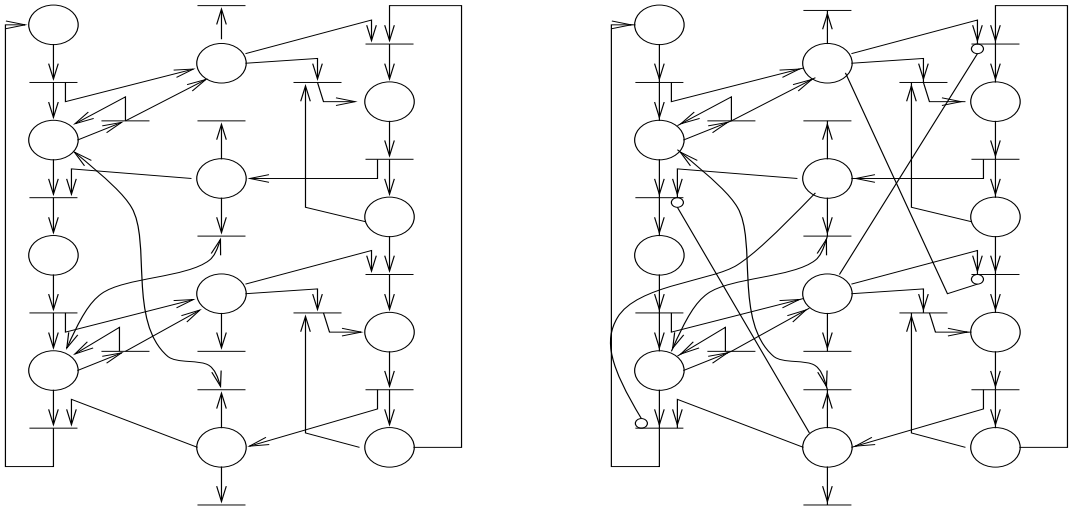


Figura 4.8. Costruzione della GSPN del protocollo *stop-and-wait*

trasmesse. L'intervallo i è relativo alla i-esima PDU di tipo DT. Sopra l'asse si trova la finestra del trasmettitore WT mentre sotto è rappresentata la finestra del ricevitore WR (figura 4.10); entrambe le finestre hanno dimensione pari ad 1 intervallo. Quando la finestra del trasmettitore è sovrapposta all'intervallo i-esimo è in corso la trasmissione della i-esima PDU di tipo DT (o meglio, il trasmettitore sta gestendo la PDU DTi, che può essere già stata trasmessa per cui si è in attesa della PDU ACKi, o in corso di trasmissione, o non ancora trasmessa), mentre analogamente, se la finestra del ricevitore è sovrapposta all'intervallo i-esimo, è in corso la ricezione (o si è in attesa) della PDU DTi. Supponiamo che il trasmettitore stia trasmettendo la PDU DTi (figura 4.10) e che fino alla PDU DT($i-1$) siano avvenute trasmissioni e ricezioni di PDU (di tipo DT ed ACK) correttamente. Il trasmettitore trasmette la PDU DTi. Il ricevitore, quando riceve la PDU DTi sposta



(a) (b)

Figura 4.9. Costruzione della GSPN del protocollo *stop-and-wait*

la sua finestra di ricezione sull'intervallo $(i+1)$-esimo e invia la PDU ACKi al trasmettitore. Il trasmettitore quando riceve ACKi sposta a sua volta avanti di una unità la finestra di trasmissione e si appresta a trasmettere la PDU successiva.

Le due finestre sono nuovamente sincronizzate. Se il trasmettitore o il ricevitore ricevono PDU fuori finestra scartano la PDU ricevuta (il ricevitore invierà comunque un ACK per ogni DT ricevuta).

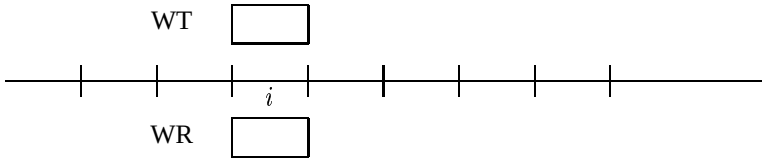


Figura 4.10. Rappresentazione delle finestre di trasmissione e di ricezione

Una rappresentazione alternativa è quella riportata nella figura 4.11, dove si suppone che le PDU siano numerate su 3 bit (da 0 a 7). L'area scura nel disegno rappresenta la posizione della finestra di ricezione e di trasmissione.

Il protocollo *stop-and-wait* in molti casi ha efficienze inaccettabili, in particolare quando il ritardo di propagazione è molto maggiore del tempo di trasmissione. Ad esempio, in un canale via satellite con le seguenti caratteristiche:

Figura 4.11. Rappresentazione alternativa delle finestre di trasmissione e ricezione

- ritardo di propagazione 270 ms (salita o discesa);
- velocità di trasmissione 50 kbit/s;
- PDU di tipo DT di 1000 bit;

il tempo di trasmissione di una PDU di tipo DT è pari a 20 ms. Un ciclo di trasmissione, cioè il tempo minimo che intercorre tra due trasmissioni, ha una efficienza (tempo di trasmissione / tempo di ciclo) pari a:

$$\frac{20}{270 + 20 + 270} = \frac{20}{560} \approx 0.035$$

che è assolutamente inaccettabile. È evidente che il protocollo *stop-and-wait* non è utilizzabile su un canale di questo tipo.

4.4 Protocollo *go-back-n*

Il primo passo per migliorare l'efficienza del protocollo *stop-and-wait* porta al protocollo a finestra detto *go-back-n*. In questo caso il trasmettitore non è costretto a fermarsi in attesa della risposta del ricevitore, lasciando il canale inutilizzato per un tempo che può essere anche molto lungo, ma è lasciato libero di trasmettere alla sua massima velocità un numero di PDU limitato, ma maggiore di 1. In altri termini, la finestra di trasmissione WT viene ad avere un'ampiezza maggiore di 1. Invece, la finestra di ricezione WR mantiene ampiezza unitaria.

Supponiamo che la finestra di trasmissione abbia ampiezza WT=2 e che la sequenza di trasmissioni inizi con DT0. Supponiamo anche che la trasmissione di DT0 e DT1 avvenga con succeso, ma che gli errori di trasmissione portino alla perdita della PDU DT2. Il ricevitore, che aspettava la PDU DT2, riceve la PDU DT3, che è fuori sequenza. Il ricevitore scarta la PDU e ritrasmette l'ACK relativo all'ultima PDU di tipo DT correttamente ricevuta, ovvero ACK1. Il trasmettitore, non avendo ricevuto ACK2, non può trasmettere DT4, perchè la sua finestra è occupata da DT2 e DT3. Allo scadere del timeout relativo a DT2 il trasmettitore ripete la trasmissione, ritornando quindi indietro di 2 posizioni (avrebbe dovuto trasmettere DT4, ma deve ripetere DT2). Il protocollo è detto *go-back-n* perché il trasmettitore torna indietro di un numero di PDU pari alla dimensione di WT.

Con questo algoritmo, se si verificano errori, il trasmettitore dovrà ritrasmettere dati già trasmessi, ma se non si verificano errori il canale è usato con buona efficienza. L'efficienza è tanto migliore quando più grande è WT.

Il funzionamento del protocollo è anche illustrato nella figura 4.12; le PDU di tipo DT fino a $i - 4$ sono state trasmesse, ricevute correttamente ed è stato ricevuto l'ACK corrispondente. La WR è posizionata sulla PDU DT($i - 1$). Nella WT si distinguono:

- PDU di tipo DT in attesa di ACK;
- PDU in propagazione;
- una PDU in trasmissione;
- PDU da trasmettere.

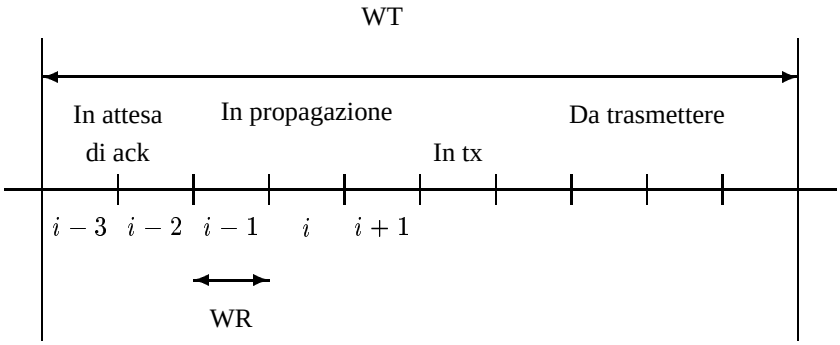


Figura 4.12. Rappresentazione delle finestre di trasmissione e ricezione per il protocollo *go-back-n*

La WR avanza di una posizione ogni volta che si riceve una PDU di tipo DT in sequenza e coretta. La WT avanza di una posizione ogni volta che si riceve una PDU di tipo ACK relativa alla posizione più bassa della finestra.

Il trasmettitore può riempire la finestra di trasmissione, ma poi deve aspettare che arrivino gli ACK.

Se il trasmettitore riceve un ACKn relativo ad una PDU DTn che non è nella posizione più bassa di WT può avanzare la WT fino alla posizione n, ignorando il fatto che alcuni ACK non sono ancora stati ricevuti (questi ACK potrebbero anche essere stati persi sul canale).

Se la numerazione delle PDU fosse illimitata, non ci sarebbero vincoli sulla dimensione della WT. Consideriamo però una numerazione su 3 bit, che permette di distinguere 8 PDU. Ovviamente la dimensione della WT non può essere superiore a 8, poiché non è accettabile avere due PDU con lo stesso numero nella stessa finestra; non saremmo infatti in grado di distinguere gli ACK relativi alle due PDU. Non è accettabile neppure una WT di dimensione pari a 8. Infatti, supponiamo di avere inviato tuttle PDU da DT0 a DT7. Il trasmettitore riceve ACK0. Sposta la WT avanti di uno e trasmette il dato DT0. Riceve nuovamente ACK0. Non è in grado di sapere se questo è un ACK cumulativo per tutte le PDU trasmesse o se è una segnalazione di ricezione fuori sequenza per cui dovrebbe ritrasmettere tutte le PDU a partire da DT1.

La lunghezza massima della finestra in trasmissione è quindi pari a $2^K - 1$, dove K è il numero di bit utilizzati per la numerazione dei dati.

4.5 Protocollo *selective repeat*

Una naturale estensione del protocollo *go-back-n* è il protocollo *selective repeat*; questo è un protocollo a finestra in cui entrambe le finestre in trasmissione ed in ricezione hanno dimensione maggiore di 1. Il ricevitore è in grado quindi di ricevere PDU non in sequenza. Con questo protocollo si ottiene una efficienza maggiore che con il protocollo *go-back-n,* al prezzo di una maggiore quantità di memoria e di una elaborazione supplementare sulle PDU.

Cerchiamo di descrivere il funzionamento del protocollo facendo riferimento al diagramma temporale della figura 4.13. Supponiamo che la PDU DT4 vada persa. Il ricevitore accetterà le PDU DT5, DT6, DT7, anche se non ha ricevuto la PDU DT4, inviando sempre ACK3 come riscontro, in quanto DT4 non è ancora stato ricevuto. Scatta il timeout del trasmettitore relativo a DT4. Il trasmettitore ritrasmette solo DT4. Il ricevitore riceve DT4 e sposta avanti la sua finestra WR fino a DT7, trasmettendo ACK7, ovvero un ACK cumulativo.

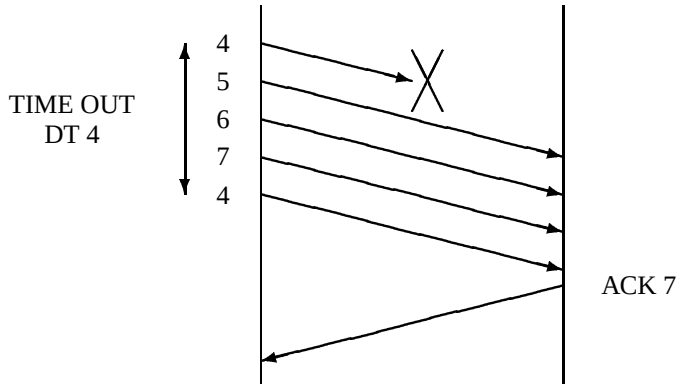


Figura 4.13. Diagramma temporale: protocollo *selective repeat*

Il vantaggio in termini di prestazioni rispetto al protocollo *go-back-n* è significativo se WR è sufficientemente grande.

Il vincolo sulle dimensioni delle finestre è ora tale per cui $(WR + WT) \leq 2^K$.

Concludiamo questo capitolo con una osservazione sui protocolli a finestra descritti finora. Il protocollo *go-back-n* è molto utilizzato poiché ha una efficienza di gran lunga maggiore rispetto al protocollo *stop-and-wait* e richiede una elaborazione supplementare minima. Il protocollo *selective repeat* offre una efficienza moderatamente maggiore rispetto al protocollo *go-back-n,* ma i prezzi da pagare in termini di memoria e di elaborazione supplementare sono tali che spesso si preferisce utilizzare comunque il protocollo *go-back-n*.

5

Livello collegamento - sottolivello MAC

Alcuni tipi di reti non utilizzano canali punto-punto, ma canali ad accesso multiplo e diffusione circolare (canali *broadcast)*. Esempi di reti siffatte si incontrano tra le reti locali (LAN – *local area networks)*, le reti metropolitane (MAN – *metropolitan area networks)*, le reti su canali radio (PRN – *packet radio networks)* e le reti via satellite (PSN – *packet satellite networks)*.

In tutti questi casi gli utenti devono essere coordinati per condividere in modo efficace l'unico canale disponibile. Le regole che disciplinano la condivisione del canale tra gli utenti costituiscono il "protocollo di accesso multiplo", che nel modello di riferimento OSI viene collocato immediatamente sopra al livello fisico, nel sottolivello 2.1, detto Medium Access Control (MAC). Si parla quindi di protocolli di sottolivello MAC o più semplicemente di protocolli MAC.

La collocazione dei protocolli di accesso multiplo nel modello di riferimento OSI non è stata del tutto indolore. Si tenga presente che da un lato i protocolli di accesso multiplo in molti casi operano direttamente sui bit in transito sul canale e quindi interferiscono con aspetti che più propriamente sarebbero specifici del livello fisico; dall'altro lato i protocolli di accesso multiplo spesso affrontano aspetti di indirizzamento e controllo di congestione che sarebbero più propriamente di pertinenza del livello rete. Questo secondo fatto è inevitabile, visto che nel caso di canali broadcast la rete di fatto coincide con il canale.

5.1 Multiplazione ed accesso multiplo

Prima di discutere dei protocolli di accesso multiplo è opportuno chiarire bene la differenza tra tecniche di *accesso multiplo* e di *multiplazione*. Entrambe le tecniche hanno come obiettivo la condivisione di un unico canale trasmissivo tra più utenti; mentre però nel caso della multiplazione tutti i flussi informativi che devono condividere il canale sono fisicamente disponibili in un medesimo sito, nel caso dell'accesso multiplo le informazioni da inviare sul canale sono fisicamente lontane. Dato quindi il problema della condivisione di un unico canale trasmissivo tra più utenti, la multiplazione costituisce la soluzione centralizzata del problema, mentre l'accesso multiplo ne costituisce la soluzione distribuita.

Le regole che definiscono gli algoritmi distribuiti (con le relative temporizzazioni ed i relativi formati) seguiti da utenti fisicamente lontani per condividere un unico canale ad accesso multiplo e diffusione circolare costituiscono il protocollo di accesso multiplo.

Le tecniche di multiplazione presuppongono che tutta l'informazione da trasmettere venga raccolta in un unico punto e poi trasmessa sul canale. La multiplazione è quindi una operazione implementata in modo centralizzato da un unico gestore del canale. Le tre tecniche di multiplazione più utilizzate sono:

- FDM, multiplazione a divisione di frequenza,
- TDM, multiplazione a divisione di tempo,
- CDM, multiplazione a divisione di codice.

Con la tecnica di multiplazione FDM si divide la banda disponibile sul canale tra gli utenti e ogni porzione di banda è assegnata in modo statico ad un utente. Tra le bande viene mantenuto un intervallo di guardia per evitare interferenza tra le trasmissioni di utenti diversi. Le trasmissioni si sovrappongono nel tempo ma sono separate in frequenza.

Con la tecnica di multiplazione TDM si suddivide il tempo in trame di lunghezza fissa e si assegna un intervallo di tempo in ogni trama a ogni utente. Tra gli intervalli viene mantenuto un tempo di guardia per evitare interferenza tra le trasmissioni di utenti diversi. Le trasmissioni sono fisicamente separate nel tempo e ogni utente può sfruttare tutta la banda del canale.

Con la tecnica di multiplazione CDM si codificano i bit da trasmettere con codici opportuni e si sfruttano le proprietà di ortogonalità di tali codici per ottenere una correlazione nulla tra le trasmissioni dei diversi utenti. Le trasmissioni si sovrappongono nel tempo e in frequenza, ma sono codificate in modo tale da essere distinguibili.

In reti a circuito, il problema dell'accesso multiplo ha trovato soluzioni simili a quelle della multiplazione: si sono infatti definiti schemi di accesso quali il TDMA (accesso multiplo a divisione di tempo) ed il FDMA (accesso multiplo a divisione di frequenza); un circuito tra due utenti viene stabilito allocando su richiesta una parte della risorsa (slot nella trama per il TDMA, o banda di frequenze per il FDMA) agli utenti che la possono utilizzare in modo continuato ed esclusivo sino al termine della comunicazione. Si noti che la distribuzione geografica della risorsa trasmissiva e delle entità comunicanti consente di far coesistere negli stessi istanti di tempo e nelle stesse bande di frequenza comunicazioni tra coppie diverse di utenti per le quali i cammini seguiti nella rete siano spazialmente disgiunti. Questo consente una condivisione della rete da parte di più utenti secondo una divisione di spazio.

Sebbene non si debba confondere l'accesso multiplo con la multiplazione, la distinzione tra le due tecniche in reti a circuito non è evidente nella fase di trasferimento dell'informazione, in quanto ogni utente utilizza una risorsa dedicata.

Nelle reti a pacchetto, poiché il traffico è di tipo impulsivo, le tecniche di multiplazione o di accesso multiplo non devono fornire un'allocazione statica delle risorse, poichè un loro uso saltuario porterebbe a gravi inefficienze (i problemi sono analoghi a quelli che hanno portato alla definizione della commutazione di pacchetto, piuttosto che ad una utilizzazione della commutazione di circuito).

In questo tipo di reti la multiplazione avviene con una tecnica chiamata multiplazione statistica a divisione di tempo (TDM statistico). Internamente al multiplatore si associa una coda ad ogni canale di uscita; i pacchetti vengono inseriti nelle code e trasmessi in successione. Il risultato è una multiplazione (perché lo stesso canale è utilizzato da più utenti) nel tempo (i pacchetti sono trasmessi in intervalli di tempo disgiunti) non fissa, ma statistica, in quanto governata dalla statistica dei flussi di pacchetti.

In presenza di traffico impulsivo, il TDM statistico ha prestazioni molto migliori del TDM tradizionale (fisso). Per renderci conto di ciò possiamo considerare il fatto che, dovendo multiplare le trasmissioni di N utenti, il comportamento del TDM statistico può essere modellato con una coda con un unico servitore a velocità 1 e con un'unica fila di attesa in cui vengono inseriti tutti i pacchetti da trasmettere, indipendentemente dalla loro provenienza. Invece, il TDM tradizionale può essere modellato con N code, ciascuna con un servitore a velocità $1/N$ e con una fila di attesa separata in cui vengono inseriti solo i pacchetti generati da un utente tra gli N. È noto il fatto che le prestazioni del sistema a servitore singolo sono vantaggiose in termini del ritardo dei pacchetti per un fattore pari al numero di utenti N.

Nel caso di reti a pacchetto con canali broadcast, i protocolli di accesso multiplo cercano di ottenere risultati simili a quelli che il TDM statistico raggiunge in un ambiente centralizzato; la distribuzione spaziale

degli utenti comporta una perdita di efficienza, per cui le prestazioni dei protocolli di sottolivello MAC sono sempre inferiori a quelle del TDM statistico, a parità di traffico.

5.2 Classificazione dei protocolli MAC

Non esiste una classificazione universalmente accettata dei protocolli MAC, però è possibile identificare alcune caratteristiche importanti. Esiste una prima distinzione tra protocolli MAC, che differenzia protocolli:

- ad accesso ordinato;
- ad accesso casuale.

I primi cercano di ottenere in un ambiente distribuito un funzionamento simile a quello che il TDM statistico ottiene in un ambiente centralizzato, spendendo una certa quantità di risorse per distribuire informazione al fine di organizzare un accesso ordinato degli utenti, evitando le interferenze. I secondi si basano sul principio di minimizzare lo spreco di risorse per coordinare l'operato degli utenti; in tal modo non si garantisce l'assenza di interferenze: quando queste si verificano, occorre porvi rimedio a posteriori.

Tra i protocolli ad accesso ordinato si possono distinguere quelli con:

- controllo centralizzato;
- controllo distribuito.

I protocolli ad accesso ordinato con controllo centralizzato sono quelli che maggiormente si avvicinano al TDM statistico, in quanto si trasporta all'unico controllore centrale tutta l'informazione sullo stato della rete per creare una coda globale delle richieste di accesso al canale. I protocolli ad accesso ordinato con controllo distribuito invece cercano di creare ad ogni utente una copia locale di una ideale coda globale delle richieste di accesso al canale, sulla base delle informazioni disponibili localmente.

Il vantaggio offerto dal controllo centralizzato consiste nel fatto che, essendovi un'unica coda, questa può essere corretta od errata (a causa di errori), ma mai inconsistente (dato che è unica). Quindi con il controllo centralizzato non vi possono essere interferenze tra due stazioni che cercano di trasmettere nello stesso momento. Tuttavia la necessità di portare tutta l'informazione sullo stato della rete ad un unico controllore può portare a perdite di efficienza tali da rendere inaccettabili le prestazioni, specialmente in canali per cui il tempo di propagazione da un estremo all'altro (*end-to-end)* sia significativo (le richieste devono essere portate al gestore, che deve distribuire il risultato dell'allocazione e finalmente l'informazione può essere trasmessa; ciò comporta un triplice attraversamento del canale, anche nel caso di traffico molto leggero).

Viceversa, con il controllo distribuito si ha una maggiore efficienza dovuta al fatto che non si deve più distribuire il risultato dell'allocazione alle varie stazioni (quindi bastano due attraversamenti del canale), ma, a causa di possibili errori, le versioni locali della coda globale possono risultare inconsistenti, quindi si possono avere interferenze o collisioni tra stazioni.

5.3 Il protocollo PODA

PODA (*Priority Oriented Demand Assignment*) è un protocollo di accesso ordinato con controllo distribuito. L'asse dei tempi, come si vede dalla figura 5.1, è suddiviso in trame di durata fissa; ogni trama a sua volta è suddivisa in un campo dati e un campo richieste. Nel campo dati possono essere inserite informazioni relative al traffico stream (isocrono) e al traffico burst (pacchetto).

Esaminiamo dapprima il caso più semplice, in cui il traffico stream non è presente. Il campo richieste può essere gestito:

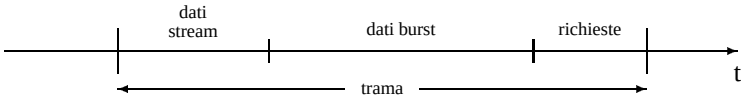


Figura 5.1. Trama utilizzata dal protocollo PODA

- con un TDM fisso (protocollo F-PODA);
- con un protocollo ad accesso casuale (protocollo C-PODA).

Esaminiamo il caso del protocollo F-PODA. Ogni utente invia le richieste di allocazione nello spazio a lui riservato nel campo richieste. Il campo richieste viene letto da tutti gli utenti, e ogni utente si crea una propria versione della coda delle richieste fatte da tutti gli utenti. Se tutte le versioni della coda sono uguali (cioè se tutti gli utenti hanno ricevuto ed elaborato correttamente tutte le richieste), ogni utente sa esattamente qual è il suo turno di trasmissione e non si verificano interferenze. Se invece, per qualche motivo, le versioni locali della coda ai diversi utenti sono diverse tra loro, prima o poi si verificano interferenze tra le trasmissioni di utenti che hanno versioni diverse.

L'organizzazione delle code delle richieste di trasmissione nel protocollo PODA è basata su priorità; ogni richiesta di accesso specifica:

- l'indirizzo sorgente;
- la durata della PDU da trasmettere;
- il ritardo massimo tollerabile prima dell'accesso al canale;
- la priorità.

Le versioni locali della coda sono ordinate secondo il ritardo massimo tollerabile e nel caso di parità secondo la priorità. Se non si riesce a trasmettere una PDU entro il ritardo massimo, questa viene eliminata dalla coda; se si può scegliere tra più PDU, si scarta quella a priorità più bassa.

In ogni PDU trasmessa, il campo dati è preceduto da un campo di intestazione (header o, in terminologia OSI, PCI), che può contenere conferme di avvenute ricezioni (ACK), ma anche richieste di accesso. Poiché ogni utente può inserire le richieste anche nell'intestazione del campo dati, spesso si preferisce gestire il campo richieste con un protocollo ad accesso casuale piuttosto che con una tecnica TDM (si usa il protocollo S-ALOHA, di cui parleremo più avanti).

L'algoritmo descritto è utilizzato per il traffico burst, ma, come accennato all'inizio del paragrafo, il campo dati della trama può trasportare anche traffico di tipo stream, ovvero sequenze di PDU di lunghezza costante, generate periodicamente. Nel caso di traffico stream le richieste riguardano l'allocazione di un circuito virtuale; in altre parole si prenota la trasmissione di una PDU di una data lunghezza in ogni trama (o di n PDU ogni trama, o di una PDU ogni k trame). Ogni utente mantiene localmente una tabella che contiene le informazioni relative ai circuiti virtuali allocati nelle trame. Quando la comunicazione è terminata, è necessario rilasciare il circuito virtuale per liberare risorse sul canale trasmissivo.

Il protocollo descritto per il traffico burst fornisce un significativo esempio di un tentativo di avvicinarsi al comportamento del TDM statistico in un ambiente distribuito. Una parte delle risorse disponibili (il campo richieste) è utilizzata per distribuire l'informazione sullo stato del sistema. Se le informazioni sono distribuite ed elaborate da tutti in modo corretto, si ha lo stesso funzionamento che si avrebbe con un gestore centralizzato; diversamente si producono interferenze.

Con un algoritmo di questo tipo è necessario che ogni stazione, quando viene accesa, si crei:

- una tabella per il traffico stream;
- un'immagine della coda globale per il traffico burst.

Per costruire la versione locale della coda relativa al traffico burst è sufficiente osservare il canale per un tempo T pari al massimo valore ammesso per il ritardo massimo tollerabile da una PDU, inserendo nella coda le richieste, ed eliminando le richieste soddisfatte. Infatti le richieste fatte prima dell'inizio dell'intervallo o sono state soddisfatte o sono state scartate.

Per il traffico stream la situazione è analoga; infatti è previsto che le stazioni possano trasmettere

- una PDU ogni trama;
- n PDU ogni trama;
- una PDU ogni k trame.

Evidentemente il terzo caso è quello che richiede il tempo di osservazione del canale più lungo. Ne consegue che è necessario per costruire l'immagine della tabella degli stream osservare il canale per un numero di trame pari a K, il massimo valore ammesso per k.

Esaminiamo in dettaglio il comportamento di un utente dal momento in cui diventa attivo. Inizialmente ogni utente è in uno **stato di acquisizione** in cui osserva il traffico sul canale per acquisire le informazioni necessarie alla creazione della tabella degli stream e della coda delle trasmissioni del traffico burst e non è autorizzata a trasmettere informazioni.

Dopo un tempo pari al massimo tra il valore T visto in precedenza e la durata di K trame, l'utente ha costruito una versione locale della coda e della tabella. Purtroppo non esiste la garanzia che tutte le informazioni siano state ricevute ed elaborate correttamente, quindi, al termine del tempo di acquisizione, si entra in uno **stato di fuori sincronismo**. In questo stato non si possono ancora trasmettere informazioni. Si trasmettono invece alcune richieste di trasmissione "finte" e si ascolta il canale quando si dovrebbe trasmettere, per controllare che nessuno trasmetta negli intervalli riservati alla stazione. Se il canale resta libero, si ha una conferma del fatto che l'immagine locale della coda e della tabella sono corrette. Se N tentativi consecutivi hanno successo (nel senso che il canale risulta libero negli intervalli prenotati), l'utente entra nello **stato di sincronismo acquisito**, cioè nello stato di funzionamento normale; in caso contrario ritorna nello stato di acquisizione. È evidente che nello stato di fuori sincronismo è necessario continuare ad aggiornare la coda e la tabella.

Durante il funzionamento normale può accadere che alcune richieste o non vengano ricevute o non vengano elaborate correttamente da qualche utente. Nel caso in cui ciò avvenga, prima o poi si verifica una collisione, perché l'utente con la tabella o la coda sbagliata trasmetterà in un istante in cui non è autorizzato. In questo caso tutte le stazioni coinvolte nella collisione si portano nello stato di fuori sincronismo e controllano che le loro informazioni sullo stato del sistema siano corrette. Quelle incolpevolmente coinvolte nella collisione verificano subito che gli intervalli di tempo a loro riservati rimangono vuoti e rientrano nello stato di sincronismo acquisito. L'utente con un'immagine errata della coda o della tabella entra invece nello stato di acquisizione ed esegue nuovamente l'algoritmo di acquisizione del sincronismo.

5.4 Il protocollo ALOHA

Il protocollo ALOHA è il capostipite della famiglia dei protocolli ad accesso casuale e deve il suo nome al fatto di essere stato usato per la prima volta presso l'università delle Hawaii per collegare con una rete via radio a pacchetto le diverse sedi universitarie sparse nell'arcipelago.

Nel caso del protocollo ALOHA gli utenti sono lasciati liberi di trasmettere senza alcun coordinamento. Può accadere che, anche in assenza di coordinamento, le trasmissioni abbiano successo, specie se il traffico sul canale è basso, quando difficilmente i segnali trasmessi da due sorgenti diverse si trovano simultaneamente sul canale ed interferiscono. Nel caso di successo si raggiunge l'obiettivo di trasmettere informazioni con un costo di controllo nullo. Questo è il meglio che si possa sperare di ottenere.

Non si può però ignorare la possibilità di interferenza (che in questo caso viene chamata *collisione*) ed è quindi necessario predisporre un insieme di meccanismi per gestire le interferenze, quando queste si verificano. Supponiamo che gli utenti siano in qualche modo in grado di riconoscere le collisioni.

Nel caso in cui le trasmissioni di due utenti collidano, entrambe devono essere ripetute. Le ritrasmissioni non possono avvenire dopo un ritardo fisso, in quanto l'interferenza si riproporrebbe sicuramente. Si deve quindi o aspettare per un tempo fisso ma diverso per ogni stazione, o scegliere un valore casuale del ritardo prima della ripetizione. La ripetizione dopo un tempo fisso ma diverso per ogni stazione ha lo svantaggio di penalizzare le stazioni con il ritardo più lungo. Per questo motivo, il protocollo ALOHA prevede la ritrasmissione dopo un ritardo casuale.

5.4.1 Efficienza del protocollo ALOHA

Data la grande semplicità del protocollo, è molto interessante studiare quali livelli di efficienza è possibile ottenere con esso. Per fare ciò è necessario costruire un modello che permetta di calcolare la frazione di tempo utilizzata per la trasmissione di PDU senza collisione.

Per un'analisi accurata dell'efficienza del protocollo ALOHA sarebbe necessario considerare un modello spazio-temporale come quello mostrato nella figura 5.2, dove si considera una sola coordinata spaziale in un sistema comprendente un satellite e due stazioni di terra. Poichè però tale modello risulta complesso, si preferisce considerare il caso di una topologia a stella (che è quella reale per esempio nel caso delle reti via satellite) e studiare il fenomeno delle collisioni al centro stella con un modello monodimensionale in cui si considera soltanto l'asse temporale.

Introduciamo due ipotesi semplificative:

- la sequenza degli istanti di tempo in cui iniziano le trasmissioni delle PDU (nuove o ripetute) forma un processo di Poisson a tasso γ [PDU/s],
- la durata delle PDU è fissa, pari a τ [s].

Calcoliamo la probabilità di successo $P_s(t_0)$ di una PDU la cui trasmissione ha inizio all'istante t_0. Tale PDU può collidere con PDU la cui trasmissione è iniziata precedentemente, in un intervallo di tempo compreso tra $t_0 - \tau$ e t_0, e con PDU la cui trasmissione inizierà successivamente, in un intervallo di tempo compreso tra t_0 e $t_0 + \tau$. $P_s(t_0)$ può quindi essere calcolata come la probabilità che il processo di Poisson a tasso γ non produca alcun inizio di trasmissione nell'intervallo $[t_0 - \tau, t_0 + \tau]$ di durata 2τ (detto *intervallo di vulnerabilità*). Si ottiene quindi

$$P_s = \mathrm{e}^{-2\gamma\tau}$$

indipendentemente da t_0, poiché l'istante di inizio della trasmissione è ininfluente ai fini del calcolo svolto.

Possiamo introdurre degli indici di misura del traffico normalizzati. Definiamo il traffico offerto G come il prodotto tra la velocità del processo di Poisson secondo cui si susseguono gli inizi delle trasmissioni delle PDU e la durata delle PDU

$$G = \gamma\tau$$

Il traffico offerto misura il carico totale del canale in PDU trasmesse per tempo di trasmissione di una PDU.

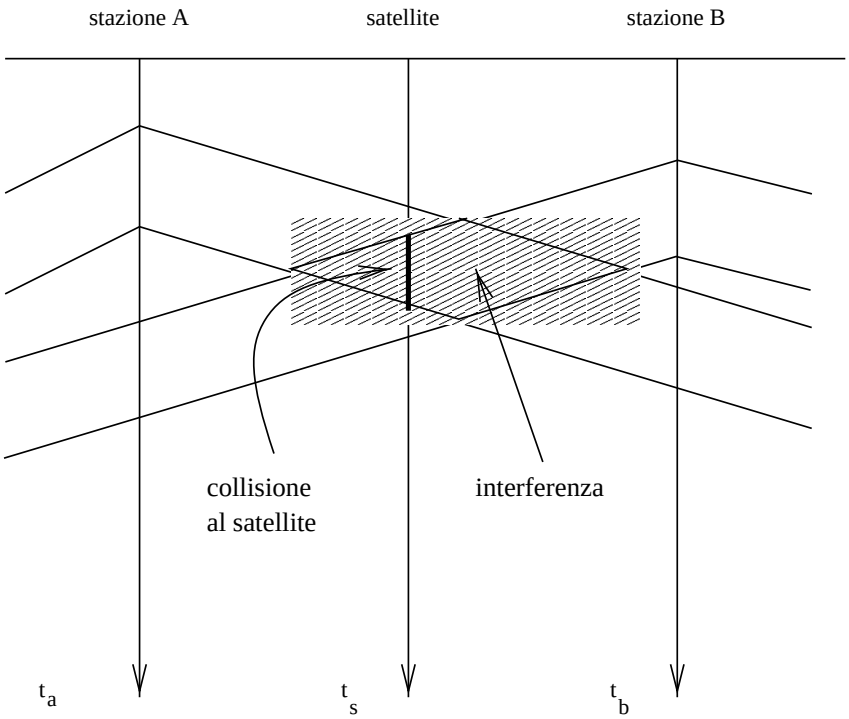


Figura 5.2. Diagramma spazio-temporale per l'analisi dell'efficienza del protocollo ALOHA

Definiamo il traffico smaltito o *throughput* S come il prodotto tra il traffico offerto G e la probabilità di successo di una PDU

$$S = GP_s = G\mathrm{e}^{-2\gamma\tau} = G\mathrm{e}^{-2G}$$

Il throughput misura la quantità di informazione effettivamente trasferita dal canale in PDU trasmesse con successo per tempo di trasmissione di una PDU.

La normalizzazione fa sì che per il throughput si abbia $0 \leq S \leq 1$, mentre per il traffico offerto $0 \leq G \leq \infty$.

In condizioni di regime, la quantità di nuovo traffico immessa nel sistema deve coincidere con la quantità di traffico smaltita. Rappresentiamo questa condizione con lo schema a blocchi della figura 5.3. La condizione di regime ci permette di scrivere la relazione tra G e S all'uscita del sistema.

Nella figura 5.4 è riportato l'andamento del throughput S in funzione del traffico offerto G. Derivando la funzione $S(G)$ rispetto a G ed uguagliando a zero si ottiene che il massimo throughput, pari a

$$S_{\max} = \frac{1}{2\mathrm{e}} \approx 0.18$$

si verifica per $G = 1/2$. La massima utilizzazione del canale è quindi solo del 18% e si ottiene con un traffico offerto G pari a metà della capacità del canale.

Esaminiamo con attenzione la curva riportata nella figura 5.4. Ovviamente per G=0 il throughput è nullo (se non si trasmette non si può avere successo). Per un traffico offerto crescente, ma inferiore a 0.5, il traffico smaltito cresce; per valori di G piccoli, poiché si verificano poche collisioni, la curva è vicina alla retta $S = G$; poi se ne discosta, ma la derivata rimane positiva. Per valori del traffico offerto maggiori di $G = 0.5$ si ha una situazione di instabilità perchè al crescere del traffico offerto il traffico smaltito diminuisce. Ciò è

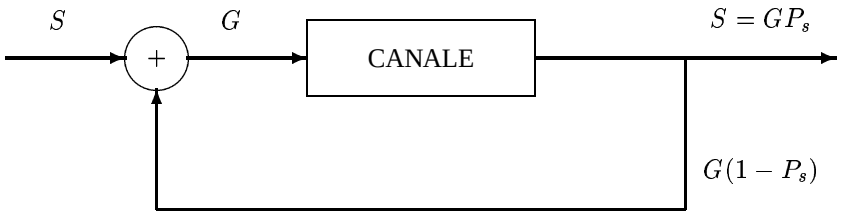


Figura 5.3. Schema del modello del protocollo ALOHA

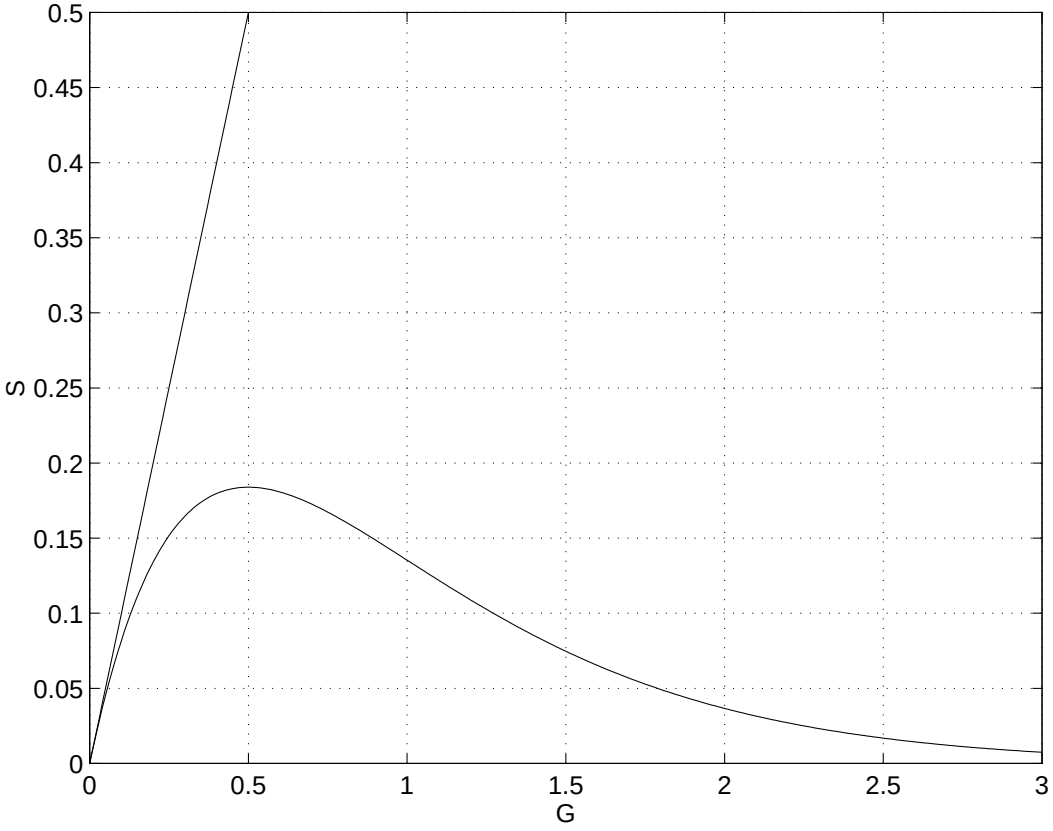


Figura 5.4. Andamento del throughput del protocollo ALOHA in funzione del traffico offerto

dovuto all'aumentare del numero di collisioni che riduce la probabilità di successo più di quanto non aumenti il traffico offerto. Ciò comporta una reazione positiva per cui rapidamente si tende ad avere un traffico G molto elevato, $S \approx 0$ e $P_s \approx 0$.

Il protocollo ALOHA è quindi intrinsecamente instabile: un momentaneo sovraccarico del canale può

innescare una reazione positiva che blocca completamente la rete. Non si può quindi operare con valori di throughput vicini al valore critico.

Esistono diverse varianti del protocollo ALOHA che regolano la velocità di trasmissione sul canale in modo da ottenere un protocollo stabile.

5.5 Il protocollo S-ALOHA

Nel protocollo S-ALOHA (Slotted-ALOHA) l'asse dei tempi è suddiviso in intervalli di durata costante, detti slot, e la dimensione delle PDU è tale per cui una PDU può essere trasmessa completamente in uno slot. Il meccanismo d'accesso è casuale, come nel protocollo ALOHA, con l'unica variante che gli utenti devono sincronizzare l'inizio della trasmissione con quello dello slot.

L'introduzione di una sincronizzazione su tutta la rete può essere difficile per la dispersione geografica degli utenti, e ha sicuramente dei costi per l'aumentata complessità delle apparecchiature. Tale aumento di complessità nel protocollo di accesso porta però a significativi vantaggi in termini di prestazioni, come si vede impostando uno studio del traffico smaltito dal protocollo.

Anche in questo caso si suppone che le PDU abbiano una durata costante τ, pari alla durata di uno slot (trascuriamo i tempi di guardia necessari per la imperfetta sincronizzazione degli utenti) e che la sequenza delle richieste di trasmissione delle PDU segua un processo di Poisson con parametro γ.

In ogni slot si possono verificare tre eventi:

- nessuna PDU trasmessa;
- una PDU trasmessa;
- più di una PDU trasmessa.

Data l'ipotesi di richieste di trasmissione di PDU secondo un processo di Poisson con parametro γ, si possono valutare facilmente le seguenti probabilità: P_0, probabilità di avere uno slot vuoto, P_1, probabilità di avere uno slot con una sola trasmissione e P_n, probabilità di avere uno slot con due o più trasmissioni.

$$P_0 = \mathrm{e}^{-\gamma\tau}$$

$$P_1 = \gamma\tau\mathrm{e}^{-\gamma\tau}$$

$$P_n = \sum_{i=2}^{\infty} \frac{(\gamma\tau)^i}{i!}\mathrm{e}^{-\gamma\tau} = 1 - P_0 - P_1 = 1 - \mathrm{e}^{-\gamma\tau} - \gamma\tau\mathrm{e}^{-\gamma\tau}$$

Il throughput S del protocollo S-ALOHA coincide con il valore medio del numero di PDU trasmesse con successo in uno slot. Poichè il numero di PDU trasmesse con successo in uno slot è una variabile casuale binaria che assume i valori 0 e 1 (se non si trasmette nessuna PDU oppure si tenta di trasmetterne più di una, assume il valore 0, mentre quando si trasmette una sola PDU assume il valore 1), il valor medio coincide con P_1. Si ottiene quindi

$$S = P_1 = \gamma\tau\mathrm{e}^{-\gamma\tau} = G\mathrm{e}^{-G}$$

Si può anche derivare lo stesso risultato in modo analogo a quanto fatto per il protocollo ALOHA. Per calcolare la probabilità di successo, esaminiamo la situazione in uno slot. Le collisioni in uno slot sono dovute solo alle PDU generate nello slot precedente. Infatti le richieste generate nello slot corrente daranno

origine a trasmissioni solamente nello slot successivo. La probabilità di successo è la probabilità che non si sia verificata nessuna richiesta nello slot precedente

$$P_s = \mathrm{e}^{-\gamma\tau} = \mathrm{e}^{-G}$$

Il throughput allora può essere ricavato come

$$S = GP_s = G\mathrm{e}^{-G}$$

Si può osservare che le relazioni ottenute sono analoghe a quelle ricavate per il protocollo ALOHA, a meno di un fattore 2 nell'esponente. Questo perchè, grazie alla sincronizzazione introdotta, l'intervallo di vulnerabilità si è ridotto da 2τ a τ.

La curva di S in funzione di G ha un andamento analogo a quello del protocollo ALOHA puro. Il massimo si ottiene per $G = 1$ e il valore del traffico smaltito corrispondente è $1/\mathrm{e}$. Questo valore è il doppio di quello ottenuto nel caso del protocollo ALOHA. Si ottiene quindi un'efficienza che nel caso migliore è pari al 36% circa. Inoltre, ponendo $G = 1$ si ottiene

$$P_0 = 1/\mathrm{e}; \quad P_1 = 1/\mathrm{e}; \quad P_n = 1 - 2/\mathrm{e}$$

Quindi, quando $G = 1$, $1/\mathrm{e}$ slot in media sono vuoti, $1/\mathrm{e}$ slot in media sono utilizzati con successo e gli slot rimanenti contengono collisioni tra due o più PDU. Purtroppo operare con $S = 1/\mathrm{e}$ non è realistico, a causa del fenomeno di instabilità già descritto per il protocollo ALOHA. È interessante osservare che, grazie alla suddivisione dell'asse dei tempi in slot e quindi all'introduzione di una sincronizzazione della rete, si riesce a raddoppiare il valore massimo teorico del traffico offerto che porta all'instabilità.

Il traffico smaltito è un indicatore parziale delle prestazioni di un protocollo; un altro parametro interessante può essere il *ritardo* con cui le PDU giungono a destinazione. Proviamo a stimare il ritardo medio delle PDU nel caso del protocollo S-ALOHA.

Poichè l'ipotesi che la sequenza degli istanti di generazione delle PDU costituisca un processo di Poisson implica che il successo della trasmissione di una PDU non dipende da quanto avviene in slot diversi da quello in cui la PDU è generata, il numero di tentativi di trasmissione di una PDU è una variabile casuale con densità di probabilità geometrica. La probabilità di effettuare k trasmissioni è quindi pari a

$$(1 - P_s)^{k-1} P_s$$

il primo fattore è dovuto ai $k - 1$ tentativi di trasmissione non andati a buon fine, il secondo al k-esimo tentativo effettuato con successo. Poiché il numero medio di trasmissioni è pari a

$$E[N_T] = 1/P_s = G/S$$

il numero medio di ritrasmissioni è pari a $G/S - 1$, valore sempre positivo perché $G \geq S$. Se consideriamo un ritardo medio di propagazione pari ad α, una durata dello slot pari a τ e un ritardo casuale di rischedulazione T_r, con valor medio $E[T_r]$, il ritardo medio totale sarà pari a:

$$E[T] = \frac{\tau}{2} + (\tau + \alpha + E[T_r]) \left(\frac{G}{S} - 1 \right) + \tau + \alpha$$

dove il secondo addendo rappresenta il ritardo che si ha quando si verifica una collisione moltiplicato per il numero di tentativi di ritrasmissione e gli altri addendi rappresentano il ritardo che si ha quando si verifica una trasmissione con successo.

5.5.1 Commento ai modelli descritti

I due modelli descritti per il calcolo dell'efficienza dei protocolli ALOHA e S-ALOHA sono basati sull'ipotesi fondamentale che il processo secondo cui si producono le richieste di trasmissione sia un processo di Poisson (lo stesso vale per gli istanti di inizio trasmissione nel caso del protocollo ALOHA, ma non nel caso dello S-ALOHA, a causa della sincronizzazione con l'inizio dello slot).

L'ipotesi poissoniana ha un'importanza fondamentale per la semplicità dei calcoli, ma ha anche implicazioni ulteriori. Infatti, tale ipotesi comporta che la probabilità di avere k richieste di trasmissione in un tempo θ è pari a

$$\frac{(\gamma\theta)^k}{k!}\mathrm{e}^{-\gamma\theta}$$

Quindi, sia nel caso del protocollo ALOHA, sia nel caso del protocollo S-ALOHA, ponendo $\theta = \tau$ si ottiene una probabilità non nulla di avere un numero arbitrariamente grande di richieste di trasmissione nel tempo necessario a trasmettere una PDU (in uno slot nel caso del protocollo S-ALOHA).

Questa osservazione ci porta a concludere che l'ipotesi poissoniana è sensata solo se la popolazione di utenti è infinita; ove il numero di utenti sia finito, infatti, supporre di avere più richieste di trasmissione che utenti in un intervallo di durata pari a τ implica un comportamento dissennato per alcune stazioni che tentano di trasmettere più di una PDU per volta, interferendo quindi con se stesse (il fenomeno è particolarmente facile da visualizzare nel caso del protocollo S-ALOHA, dove è evidente che non ha senso pensare ad un numero di richieste di trasmissione per slot maggiore del numero di utenti).

Poiché le valutazioni fatte sotto l'ipotesi di una popolazione infinita di utenti possono risultare non realistiche, nel seguito sviluppiamo due modelli basati sull'ipotesi di una popolazione limitata, nel caso del protocollo S-ALOHA.

Modello a popolazione finita del protocollo S-ALOHA: prima versione

Supponiamo che alla rete siano collegati N utenti, con N finito.

Invece di caratterizzare il processo delle richieste di trasmissione in modo globale come nel caso precedente, tenendo conto della discretizzazione dell'asse dei tempi, descriviamo con un processo di Bernoulli la successione delle richieste di trasmissione di PDU di ogni singolo utente. Il processo di Bernoulli è il processo risultante da una successione di scelte ripetute in modo indipendente ad ogni intervallo; in termini applicativi ciò implica che ogni utente ad ogni slot effettua una scelta binaria relativa alla trasmissione di una PDU nel prossimo slot. Ciò comporta una distribuzione di tipo geometrico del numero di slot tra due trasmissioni consecutive da parte di uno stesso utente. Il parametro che caratterizza il processo di Bernoulli è la probabilità di trasmissione di una PDU in uno slot arbitrario. Chiamiamo G_m la probabilità che l'utente m trasmetta una PDU in un generico slot k; supponiamo che tale probabilità non dipenda da k. Quindi, in ogni slot, l'utente m trasmette con probabilità G_m e non trasmette con probabilità $(1 - G_m)$. Poichè la variabile casuale "numero di PDU trasmesse in un generico slot dall'utente m" assume solo i valori 0 o 1, G_m può anche essere interpretato come il numero medio di PDU trasmesse dall'utente m in un generico slot, quindi come il traffico normalizzato offerto dall'utente m alla rete.

Il traffico totale offerto alla rete, G, può essere ricavato mediante la somma dei traffici offerti dai singoli utenti:

$$G = \sum_{m=1}^{N} G_m$$

e, analogamente, il traffico totale smaltito dalla rete sarà pari a

$$S = \sum_{m=1}^{N} S_m$$

dove S_m è il numero medio di PDU trasmesse *con successo* dall'utente m in uno slot arbitrario, che coincide con la probabilità che l'utente m trasmetta con successo in uno slot arbitrario, ed è quindi un numero compreso tra 0 e 1.

Viste le ipotesi fatte, si ricava che

$$S_m = G_m \prod_{j=1, j\neq m}^{N} (1 - G_j)$$

perché l'utente m ha successo in un generico slot solo se egli trasmette e gli altri $N-1$ utenti non trasmettono.

Tramite la relazione sopra, a partire dalla conoscenza dei valori di G_m per tutti gli utenti siamo in grado di calcolare S_m e quindi il throughput S.

Se tutti gli utenti si comportano nello stesso modo, ovvero $\forall m, G_m = G/N$, allora si ha

$$S_m = \frac{G}{N}\left(1 - \frac{G}{N}\right)^{N-1}$$

e quindi

$$S = G\left(1 - \frac{G}{N}\right)^{N-1}$$

Facendo tendere il numero degli utenti N ad infinito, si ricava

$$\lim_{N\to\infty} G\left(1 - \frac{G}{N}\right)^{N-1} = G\mathrm{e}^{-G}$$

ovvero si ottiene lo stesso risultato del modello a popolazione infinita.

Calcoliamo ora il valore di G che massimizza S, uguagliando a zero la derivata di S rispetto a G.

$$\frac{dS}{dG} = \left(1 - \frac{G}{N}\right)^{N-1} + G(N-1)\left(1 - \frac{G}{N}\right)^{N-2}\left(-\frac{1}{N}\right) = 0$$

Dopo semplici passaggi si ottiene

$$\left(1 - \frac{G}{N}\right)^{N-2}(1 - G) = 0$$

che fornisce le due soluzioni $G = 1$ e $G = N$.

Esaminando la derivata seconda si vede che per $G = N$ non si ha un massimo; si ha invece un massimo per $G = 1$ e il traffico smaltito totale in questo caso vale

$$S_{\max} = \left(1 - \frac{1}{N}\right)^{N-1}$$

D'altra parte anche un esame del significato dei due valori di G poteva permettere di concludere che il caso di interesse è $G = 1$. Infatti $G = N \Rightarrow G_m = 1\ \forall m$, e quindi ogni utente trasmette con probabilità 1 in ogni slot; in questo caso si verificano sempre collisioni multiple, di modo che sicuramente $S = 0$.

Il risultato relativo alla condizione di massimo throughput, ricavato nel caso in cui gli utenti hanno la stessa caratterizzazione probabilistica, vale in generale; è sempre vero che $S = S_{\max}$ quando $\sum_{i=1}^{N} G_i = 1$. In questa condizione viene trasmessa mediamente una PDU in ogni slot; questa è la condizione che porta alla migliore efficienza.

Esaminiamo il caso particolare di $N = 2$ utenti che hanno differente caratterizzazione probabilistica. Caratterizziamo i due utenti rispettivamente con le probabilità G_1 e G_2. In questo caso, poiché $S_1 = G_1(1 - G_2)$ e $S_2 = G_2(1 - G_1)$ si ottiene

$$S = G_1(1 - G_2) + G_2(1 - G_1)$$

Il problema della ricerca del massimo in questo caso è bidimensionale: si deve trovare il massimo valore di una funzione di due variabili. Come già detto per il caso generale, si ricava che il massimo si ha in corrispondenza di $G_1 + G_2 = 1$. Sostituendo allora $G_2 = 1 - G_1$ nell'espressione di S si ottiene

$$S_{\max} = G_1^2 + (1 - G_1)^2$$

Nella figura 5.5 è riportato l'andamento di S_{max} in funzione di G_1. Come si può osservare, il protocollo S-ALOHA ha traffico smaltito pari a 1 per $G_1 = 0$ e $G_1 = 1$; entrambi questi casi indicano la condizione in cui un utente trasmette sempre e l'altro utente non trasmette mai Se entrambi gli utenti trasmettono in modo bilanciato il throughput massimo è 0.5 (si ricordi che con una popolazione infinita il modello portava ad un traffico smaltito massimo pari al 36%).

Due osservazioni conclusive:

- le prestazioni peggiorano al crescere del numero di utenti, supponendo che la caratterizzazione probabilistica di tutti gli utenti sia identica; quindi il caso $N \to \infty$ è il peggiore;
- il caso in cui la caratterizzazione probabilistica di tutti gli utenti è identica porta alle prestazioni peggiori (per qualsiasi numero di utenti); se invece il traffico offerto dagli utenti è sbilanciato, il protocollo tende a comportarsi in modo più efficace. Questo fenomeno è dovuto al fatto che l'utente che genera una grande quantità di traffico opera a tutti gli effetti una multiplazione prima di tentare un accesso multiplo. Si può quindi affermare che un traffico sbilanciato permette al protocollo di funzionare in maniera più efficiente.

Modello a popolazione finita del protocollo S-ALOHA; seconda versione

Anche in questo caso descriviamo il protocollo S-ALOHA caratterizzando il comportamento di ogni utente in modo individuale. Adesso però supponiamo che il comportamento relativo alla trasmissione della PDU la prima volta sia diverso dal comportamento relativo alle eventuali ritrasmissioni dovute a collisione; supponiamo inoltre che ogni utente abbia un unico buffer in cui memorizzare le PDU da trasmettere.

La disponibilità di un unico buffer fa sì che la entità di sottolivello MAC che gestisce il protocollo possa essere solamente o nello stato LIBERO, perché non ha PDU da trasmettere (e quindi il buffer è libero), o nello stato OCCUPATO, perché il suo unico buffer è occupato (ed è in corso l'algoritmo per la ritrasmissione). Il comportamento di un utente nello stato LIBERO è legato alla dinamica della generazione delle PDU, mentre il comportamento nello stato OCCUPATO è legato alla dinamica della ripetizione delle trasmissioni.

Supponiamo che il sistema comprenda N utenti con identica caratterizzazione probabilistica. Un utente LIBERO riceve una richiesta di trasmissione in uno slot generico con probabilità p. Un utente OCCUPATO ripete la trasmissione della PDU presente nel buffer con probabilità α. La distribuzione del ritardo tra un tentativo di ritrasmissione e il successivo è geometrica con parametro α: quindi la probabilità che vi siano i slot tra due tentativi di ritrasmissione è pari a $(1 - \alpha)^{i-1}\alpha$.

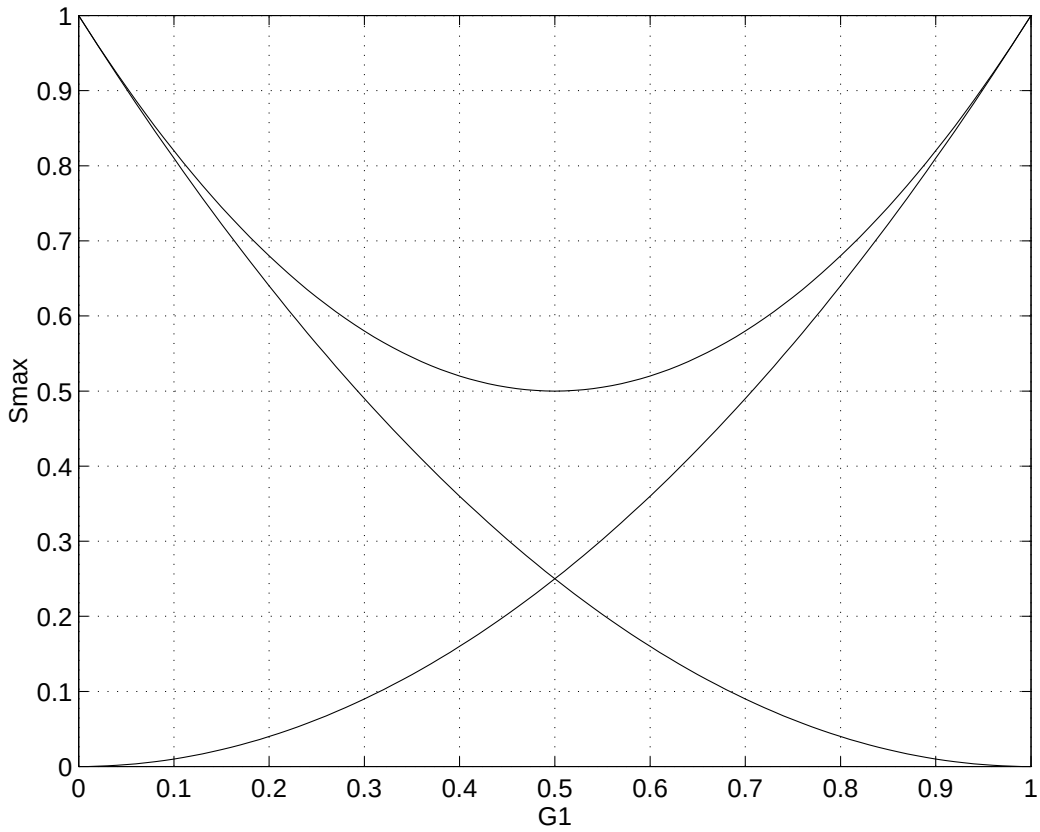


Figura 5.5. Andamento del throughput del protocollo S-ALOHA con due utenti

Questo modello può essere analizzato con una catena di Markov a tempo discreto il cui stato è definito dal numero di utenti nello stato OCCUPATO nel sistema. Un passo nel modello a tempo discreto corrisponde alla durata di uno slot.

La catena con $N = 3$ è riportata nella figura 5.6. Calcoliamo le probabilità di transizione a partire dallo **stato 0**.

- Si resta nello stato 0 perché non viene generata alcuna PDU [probabilità $(1-p)^3$], oppure perché unaPDU viene generata e trasmessa con successo da un unico utente mentre gli altri due non generano PDU [probabilità $3p(1-p)^2$].
- Si va nello stato 3 quando tutti e tre gli utenti generano una PDU e si verifica una collisione tra i tre utenti (probabilità p^3).
- Si va nello stato 2 quando due utenti generano una PDU, mentre il terzo non tenta di trasmettere. Si genera una collisione fra due utenti: [probabilità $3p^2(1-p)$].
- La probabilità di transizione dallo stato 0 allo stato 1 è zero, poiché se un solo utente trasmette, la

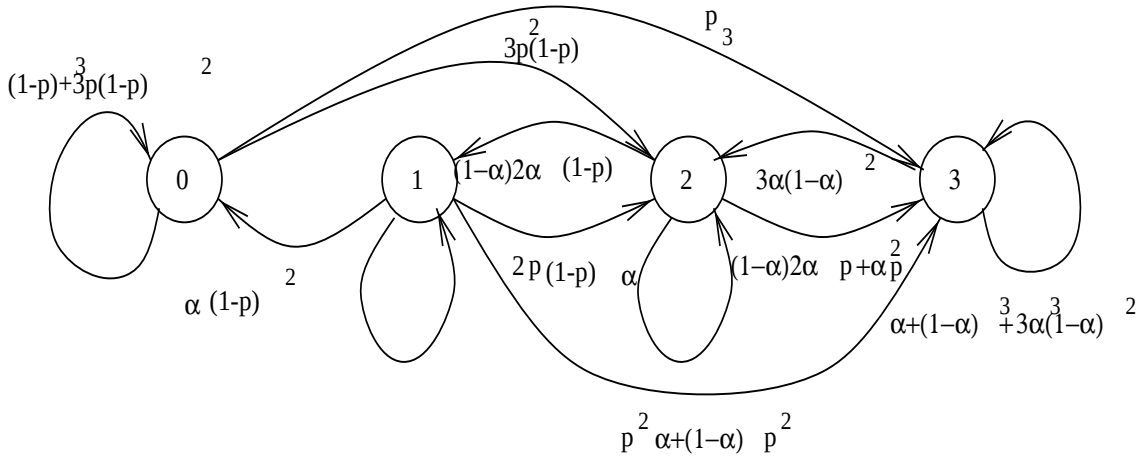


Figura 5.6. CMTD del protocollo S-ALOHA con tre utenti

trasmissione ha successo per definizione.

Si noti che le possibili combinazioni dei comportamenti dei tre utenti (trasmissione, non trasmissione) sono otto (2^3).

Calcoliamo ora le probabilità di transizione a partire dallo **stato 3**. Tutti gli utenti sono nello stato OCCUPATO; possono quindi ritrasmettere, con probabilità α, o non ritrasmettere, con probabilità $1-\alpha$.

- Si resta nello stato 3 se si produce una collisione tra due o tre utenti, oppure se nessun utente ritrasmette [probabilità $3\alpha^2(1-\alpha)+\alpha^3+(1-\alpha)^3$].
- Si va nello stato 2 se un solo utente ritrasmette e gli altri due non trasmettono [probabilità $3\alpha(1-\alpha)^2$].

Ovviamente la probabilità di andare dallo stato 3 negli stati 1 o 0 è nulla. Si noti che anche in questo caso le possibili combinazioni dei comportamenti dei tre utenti (ritrasmissione, non ritrasmissione) sono otto (2^3).

Calcoliamo ora le probabilità di transizione a partire dallo **stato 2**. In questo caso i comportamenti sono relativi alla dinamica delle ripetizioni per i due utenti nello stato OCCUPATO, ed alla dinamica delle generazioni delle richieste di trasmissione per l'utente nello stato LIBERO. Cionondimeno, le possibili combinazioni dei comportamenti dei tre utenti (ritrasmissione, non ritrasmissione) sono ancora otto (2^3).

- Si va nello stato 1 se uno degli utenti che era OCCUPATO diventa LIBERO. Ciò può avvenire se uno dei due utenti che era OCCUPATO trasmette, mentre l'altro utente che era nello stato OCCUPATO e quello che era nello stato LIBERO non trasmettono [probabilità $2\alpha(1-\alpha)(1-p)$].
- Si va nello stato 3 se trasmettono sia l'utente LIBERO sia uno dei due (o entrambi) gli utenti nello stato OCCUPATO [probabilità $2p\alpha(1-\alpha)+p\alpha^2$].
- La probabilità di restare nello stato 2 si può ricavare facilmente sottraendo le probabilità calcolate in precedenza da 1: $(1-p)(1-\alpha)^2+p(1-\alpha)^2+(1-p)\alpha^2$.

Infine le probabilità di transizione a partire dallo **stato 1** sono:

- allo stato 3: $p^2(1-\alpha) + p^2\alpha$ (i due utenti nello stato LIBERO trasmettono contemporaneamente e quello OCCUPATO indifferentemente trasmette o no);
- allo stato 2: $2p(1-p)\alpha$ (l' utente OCCUPATO e uno dei due nello stato LIBERO trasmettono);
- allo stato 0: $\alpha(1-p)^2$ (l' utente OCCUPATO riesce a trasmettere con successo in quanto gli altri due non trasmettono);
- si rimane nello stato 1 con probabilità: $(1-p)^2(1-\alpha) + 2p(1-p)(1-\alpha)$ (nessun utente trasmette oppure uno dei due utenti nello stato LIBERO trasmette con successo e gli altri utenti non trasmettono).

Per esaminare il caso generale di N utenti, definiamo due variabili casuali n e r che rappresentano rispettivamente il numero di PDU nuove trasmesse in uno slot e il numero di PDU ripetute in uno slot. Esprimiamo le probabilità di transizione p_{ij} in funzione di queste due variabili.

$$p_{ij} = \begin{cases} P\{r=1 \mid i\}P\{n=0 \mid i\} & j = i-1 \\ P\{n=1 \mid i\}P\{r \geq 1 \mid i\} & j = i+1 \\ P\{n=j-i \mid i\} & j > i+1 \\ P\{n=0 \mid i\}P\{r \neq 1 \mid i\} + P\{n=1 \mid i\}P\{r=0 \mid i\} & j = i \\ 0 & j < i-1 \end{cases}$$

Commentiamo le espressioni per le p_{ij}:

- per $j = i-1$ si è ritrasmessa una PDU con successo e nessuna nuova PDU è stata trasmessa;
- per $j = i+1$ si è tentato di trasmettere una PDU nuova e almeno una PDU ripetuta, generando così una collisione;
- per $j > i+1$ le nuove PDU trasmesse sono in numero $j-i > 2$, in modo da collidere anche in assenza di trasmissioni ripetute; quindi il numero di ritrasmissioni è irrilevante;
- per $j = i$ lo stato non muta; non si hanno nuove trasmissioni e le ritrasmissioni sono o 0 (lo slot non è stato usato) o più di una (si è verificata una collisione tra ripetizioni), oppure si è trasmessa con successo una nuova PDU e non vi sono state ritrasmissioni;
- per $j \leq i-1$ la probabilità è nulla, in quanto non si possono ripetere più trasmissioni con successo in un unico slot.

Dobbiamo ora calcolare le probabilità definite sopra. Esaminiamo inizialmente la $P\{n=k \mid i\}$, ovvero la probabilità di avere k trasmissioni nuove a partire da uno stato con i utenti occupati. Le nuove trasmissioni sono prodotte dagli $N-i$ utenti liberi, che si comportano in modo indipendente tra di loro. Quindi

$$P\{n=k \mid i\} = \binom{N-i}{k} p^k \, (1-p)^{N-i-k}$$

Analogamente

$$P\{r=k \mid i\} = \binom{i}{k} \alpha^k \, (1-\alpha)^{i-k}$$

Possiamo in generale calcolare la distribuzione stazionaria della CMTD, che è anche quella di regime poiché la catena è ergodica.

Conoscendo la distribuzione di regime si può facilmente ricavare il numero medio di utenti nello stato OCCUPATO

$$E[N_0] = \sum_{i=1}^{N} i\pi_i$$

Il traffico smaltito dalla rete, o throughput, è il numero medio di PDU trasmesse con successo in uno slot

$$S = \sum_{i=0}^{N} P_{s|i}\pi_i$$

dove $P_{s|i}$ indica la probabilità di successo condizionata dal trovarsi nello stato i. La trasmissione ha successo quando si trasmette una nuova PDU senza interferenza o quando si ritrasmette una PDU vecchia senza interferenza

$$P_{s|i} = P\{n = 1 \mid i\}P\{r = 0 \mid i\} + P\{n = 0 \mid i\}P\{r = 1 \mid i\}$$

Avendo calcolato il throughput e il numero medio di utenti nello stato OCCUPATO, grazie al teorema di Little si può calcolare il ritardo medio come rapporto tra il numero medio di utenti nello stato OCCUPATO e il throughput

$$E[T] = \frac{E[N_0]}{S}$$

Riportando in un grafico i risultati forniti dal modello, si ottengono le curve della figura 5.7 per il throughput S e per il ritardo medio $E[T]$ in funzione di p con α come parametro. Inoltre, riportando la curva di $E[T]$ in funzione di S (si veda la figura 5.8), si può notare che per un unico valore di S esistono due valori ammissibili di $E[T]$.

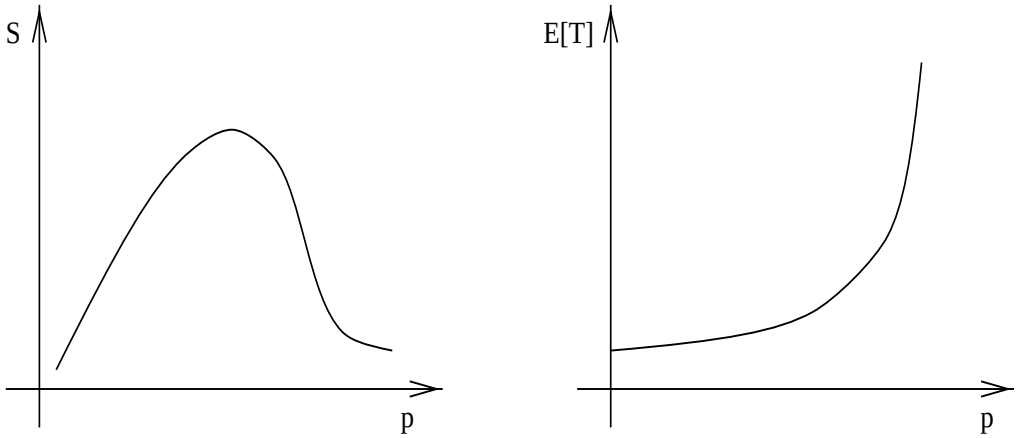


Figura 5.7. S e $E[T]$ in funzione di p

Questo modello a popolazione finita è abbastanza preciso e descrive bene il comportamento del sistema, ma nella versione esaminata ha due limiti:

- la caratterizzazione probabilistica è identica per tutti gli utenti;

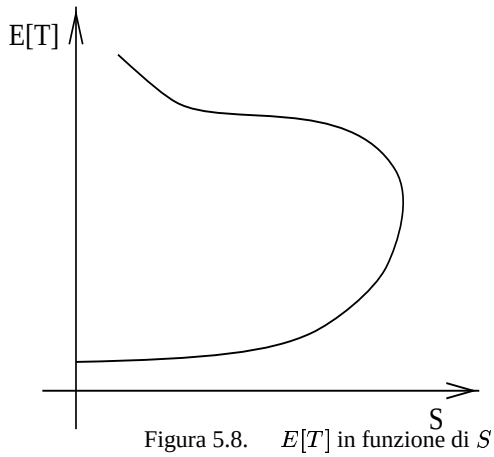


Figura 5.8. $E[T]$ in funzione di S

- la capacità di memoria di ciascun utente è limitata a una sola PDU.

Entrambe le limitazioni corrispondono ad ipotesi introdotte per rendere il modello semplice. È possibile rimuovere tali ipotesi semplificative, ma si ottengono modelli significativamente più complicati.

Ad esempio, eliminando il vincolo sull'uguaglianza della descrizione probabilistica degli utenti, lo stato deve identificare ogni utente singolarmente; quindi sono necessari 2^N stati per rappresentare il sistema (tutte le possibili combinazioni di utenti negli stati LIBERO e OCCUPATO). Di conseguenza si modificano le probabilità di transizione.

Volendo invece distinguere un solo utente da tutti gli altri (un esempio frequente è la descrizione di un *server* in una rete di calcolatori) sarebbero necessari $2N$ stati.

Per superare invece il vincolo sulla dimensione unitaria del buffer, si rappresenta lo stato di ogni utente con le le condizioni "utente libero", "utente con 1 posizione occupata", ..., "utente con M posizioni occupate"; si ottengono N^M stati.

5.6 I protocolli CSMA

Se un gruppo di persone in una sala vuole iniziare una discussione, è necessario imporre delle regole per l'utilizzazione del canale acustico:

- si può utilizzare un moderatore che decida chi deve parlare di volta in volta;
- si può utilizzare un metodo di allocazione su richiesta per alzata di mano;
- si può sperare che i partecipanti siano educati, ascoltino se qualcuno sta parlando e inizino a parlare solo quando c'è silenzio;
- si può infine lasciare che le persone parlino anche se qualcuno sta già parlando.

Approssimativamente, il primo sistema è l'equivalente di un protocollo TDMA, il secondo utilizza un canale separato di prenotazione come nel protocollo PODA, il quarto corrisponde al protocollo ALOHA. Il terzo, infine, è l'equivalente del protocollo CSMA.

Il protocollo CSMA (Carrier Sense Multiple Access) è un protocollo utilizzabile in sistemi in cui il ritardo di propagazione è breve rispetto alla durata della trasmissione di una PDU: nel seguito dell'analisi si vedrà che il ritardo di propagazione condiziona pesantemente le prestazioni del protocollo. Infatti ogni stazione prima di trasmettere verifica che il canale sia libero; se il canale è occupato, si rinvia la trasmissione. Se il ritardo di propagazione è piccolo allora l'informazione raccolta dalla stazione è significativa, altrimenti le prestazioni offerte dal protocollo CSMA possono essere addirittura peggiori di quelle del protocollo ALOHA. Questo avviene perché, a causa del ritardo di propagazione, le informazioni che la stazione ottiene dall'ascolto del canale sono "vecchie", cioè non rispecchiano la situazione istantanea della rete in ogni suo punto, ed è in questo caso preferibile non utilizzare alcuna informazione (come fa il protocollo ALOHA) piuttosto che utilizzare informazioni sbagliate. Per questo motivo il protocollo CSMA è utilizzabile solo in reti geograficamente limitate, ovvero in reti locali o in reti che usano canali radio.

Il protocollo è basato sulle seguenti regole:

- l'entità di sottolivello MAC che vuole trasmettere si mette in ascolto del canale (Carrier Sense);
- se il canale è occupato non si trasmette per non dare luogo ad una collisione, ma si ritenta in un momento successivo;
- se il canale è libero si inizia la trasmissione; la trasmissione può non andare a buon fine perché si può verificare una collisione; in questo caso si ritrasmette dopo avere atteso un tempo casuale.

Quando una stazione che deve trasmettere (o ritrasmettere) una PDU trova il canale occupato, si può comportare secondo modalità diverse, a seconda della versione del protocollo CSMA.

Se si continua ad ascoltare il canale e si inizia la trasmissione non appena il canale diventa libero, il protocollo è detto **1-persistente** (1P). Se invece si attende un tempo casuale prima di riascoltare il canale per accertarsi che sia libero, il protocollo è detto **non persistente** (NP) o **0-persistente**. Se si continua ad ascoltare il canale con probabilità p e con probabilità $(1 - p)$ si riprova dopo un tempo casuale si ha il protocollo p-**persistente**, di cui il protocollo non persistente e il protocollo 1-persistente sono i due casi estremi.

Si osservi che le collisioni possono comunque verificarsi, a causa del fatto che il ritardo di propagazione non è nullo. Supponiamo infatti che la stazione A senta il canale libero e inizi a trasmettere. La stazione B in un istante immediatamente successivo sente ancora il canale libero, poiché il segnale si sta propagando da A a B ma non è ancora giunto alla stazione B. Se anche B inizia a trasmettere si crea una collisione, distruttiva per entrambe le PDU.

Per questo motivo nel caso di protocollo non persistente l'intervallo di vulnerabilità, cioè l'intervallo di tempo in cui vi possono essere collisioni, è pari a due volte il tempo massimo di propagazione nella rete, in quanto la stazione A può accorgersi dell'avvenuta collisione solo quando le arriva il segnale prodotto dalla stazione B.

Nel caso del protocollo 1-persistente si ha un'ulteriore causa di collisione; infatti se due stazioni che utilizzano il protocollo 1-persistente decidono di trasmettere, e ponendosi in ascolto del canale si accorgono che una terza stazione sta già trasmettendo, entrambe si porranno in attesa del momento in cui il canale si libera. Quando ciò avviene entrambe iniziano a trasmettere contemporaneamente, collidendo.

Se invece vi è una sola stazione che, volendo trasmettere, trova il canale occupato, quando il canale si libera essa inizierà subito la trasmissione, senza aspettare altro tempo.

Per questo possiamo aspettarci che le prestazioni del protocollo 1P siano migliori di quelle del protocollo NP ai carichi bassi, quando la probabilità che due o più stazioni si pongano in attesa contemporaneamente è bassa, mentre il protocollo NP, grazie alla minore possibilità di collisione, sia più efficiente in corrispondenza di un traffico offerto alla rete elevato.

Studiamo le prestazioni di questo protocollo per verificare se il suo comportamento è migliore rispetto a quello del protocollo ALOHA; è evidente che il ritardo di propagazione è un parametro fondamentale di questo protocollo e ne determina le prestazioni in modo decisivo; infatti quanto più due stazioni sono vicine tanto meno è probabile che le loro trasmissioni collidano.

5.6.1 Modello per l'analisi delle prestazioni

Il modello del protocollo CSMA non persistente è abbastanza complicato perché coinvolge aspetti spaziali oltre che temporali. La figura 5.9 rappresenta un diagramma spazio-tempo, in cui in ascissa è rappresentata la distanza tra le stazioni (è sufficiente una direzione poiché la topologia normalmente utilizzata è la topologia lineare a bus) e il tempo è rappresentato come ordinata ed è crescente verso il basso.

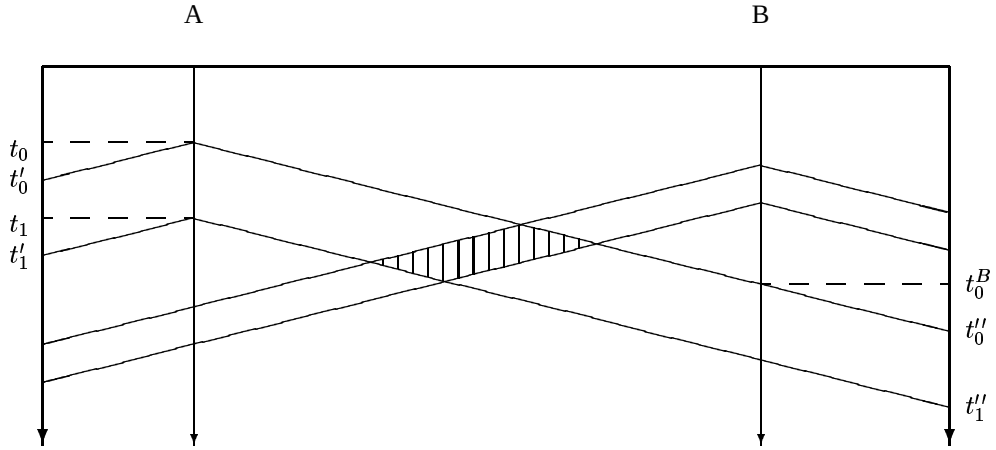


Figura 5.9. Diagramma spazio-temporale di trasmissione con collisione.

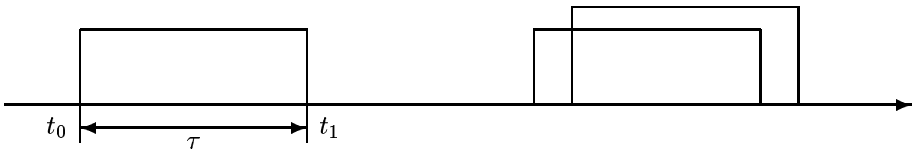


Figura 5.10. Diagramma temporale

Consideriamo due stazioni A e B e supponiamo che A inizi a trasmettere una PDU all'istante t_0. L'informazione si propaga su tutto il bus in entrambe le direzioni. La trasmissione della PDU termina all'istante t_1. Indichiamo con l'apice $'$ ($''$) gli istanti di tempo in cui l'informazione raggiunge l'estremo sinistro (destro) del bus; in generale $t'_1 \neq t''_1$. Indichiamo con t_0^B l'istante in cui la stazione B si accorge dell'inizio della trasmissione della PDU trasmessa dalla stazione A. In tutto l'intervallo $[t_0,t_0^B]$ la stazione A ha trasmesso informazione e la stazione B non se ne è ancora accorta. Se in questo intervallo B inizia la trasmissione di una PDU si ha collisione. La collisione è evidenziata nella figura come una zona di sovrapposizione delle due aree. I pezzi di PDU non disturbati (le aree che non si sovrappongono) sono comunque inutilizzati al fine della trasmissione della PDU.

Si può valutare la probabilità di collisione calcolando il valor medio delle aree in cui si ha interferenza. Il throughput può essere calcolato come il valor medio della parte del diagramma spazio-tempo occupata con trasmissioni che hanno successo.

Il calcolo svolto sul diagramma spazio-tempo è complesso, cosìsi preferisce svolgere un'analisi approssimata utilizzando solo un asse temporale. Sappiamo che la trasmissione di una PDU occupa un intervallo di tempo $[t_0,t_1]$ alla stazione A e un intervallo $[t'_0,t'_1]$ ($[t''_0,t''_1]$) all'estremo sinistro (destro) del bus. Non essendo nota la posizione della stazione A sul bus, non è immediato il posizionamento della trasmissione della PDU sull'asse temporale. Facciamo un'ipotesi conservativa, che ci consenta un'analisi pessimistica: la PDU occupa il bus per un tempo pari al suo tempo di trasmissione più il massimo ritardo di propagazione sul bus. Con questa ipotesi nella figura 5.10 sono rappresentate a sinistra la trasmissione di una PDU con successo e a destra la trasmissione con collisione.

Ipotizziamo che la trasmissione delle PDU abbia una durata costante pari a τ; tutti i tentativi di trasmissione delle PDU (le PDU trasmesse per la prima volta – con successo o collisione – quelle ritrasmesse perché si è verificata una collisione e quelle non trasmesse perché il canale era occupato) sono descritti da un processo di Poisson a parametro γ. Il ritardo di propagazione end-to-end è indicato dalla lettera δ, mentre β è una variabile casuale che varia tra 0 e δ e rappresenta il tempo trascorso tra la prima trasmissione che ha subito collisione e l'ultima trasmissione che subirà collisione.

Normalizziamo tutte le grandezze rispetto al tempo di trasmissione τ di una PDU. Il ritardo di propaga-

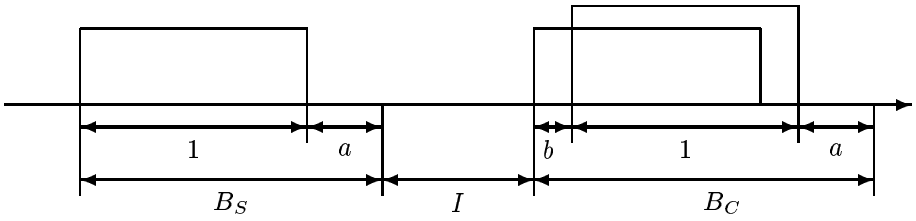


Figura 5.11. Diagramma temporale con le grandezze normalizzate.

zione normalizzato alla durata della PDU diventa $a = \delta/\tau$, γ diventa $G = \gamma\tau$ (infatti γ ha le dimensioni di s^{-1}) e $b = \beta/\tau$. Con questa normalizzazione la figura 5.10 si trasforma nella figura 5.11, dove, per definizione, $b < a$.

Sul canale si alternano intervalli in cui il canale è libero (indicati con I, da "Idle") e altri in cui è occupato (che indichiamo con B, da "Busy"). Quando il canale è occupato, è in corso una trasmissione che può avere successo o meno. Il calcolo del throughput passa attraverso il calcolo dei valori medi di I, B_s e B_c, rispettivamente tempo per cui il canale è libero, tempo per cui il canale è occupato dalla trasmissione con successo di una PDU e tempo per cui il canale è occupato dalla trasmissione di PDU che subiscono collisione.

Poiché le PDU hanno durata costante unitaria, la durata di B_s è costante e pari a: $1 + a$, per cui $E[B_s] = 1 + a$. I tentativi di trasmissione seguono un processo di Poisson con parametro G. Il valor medio di I è un tempo di ricorrenza e, per la proprietà di assenza di memoria, è pari a $1/G$; ne consegue che $E[I] = 1/G$. Resta da calcolare il valor medio di $B_c = b+1+a$; ma poiché b è una variabile casuale $E[B_c] = 1+a+E[b]$. b è il tempo che trascorre tra la prima trasmissione che subisce una collisione e l'ultima trasmissione che subirà collisione. Per valutare la densità di probabilità di b calcoliamo $P\{b \leq y \mid \text{collisione}\}$.

$$P\{b \leq y \mid \text{collisione}\} = P\{b \leq y \mid \geq 1 \text{ arrivi nell'intervallo } [0,a]\}$$

Il passaggio precedente è corretto solo perché il processo è stazionario per cui gli indici di prestazione non dipendono dall'origine dell'asse dei tempi; l'intervallo sarebbe propriamente $[t_0,t_0 + a]$.

L'evento $(b \leq y)$ in caso di collisione è equivalente all'evento (0 arrivi tra y e a), poiché b è, per definizione, l'istante dell'ultimo arrivo. Allora

$$P\{b \leq y \mid \text{collisione}\} =$$

$$= P\{0 \text{ arrivi in } [y,a] \mid \geq 1 \text{ arrivo in } [0,a]\} =$$

$$= \frac{P\{0 \text{ arrivi in } [y,a], \geq 1 \text{ arrivo in } [0,a]\}}{P\{\geq 1 \text{ arrivo in } [0,a]\}} =$$

I due intervalli $[y,a]$ e $[0,a]$ non sono disgiunti; per avere due eventi statisticamente indipendenti e potere esprimere la probabilità congiunta come prodotto di probabilità bisogna avere due intervalli separati. Poiché non ci sono arrivi tra y e a, gli arrivi nell'intervallo $[0,a]$ si possono verificare solo tra 0 e y,

$$= \frac{P\{0 \text{ arrivi in } [y,a], \geq 1 \text{ arrivo in } [0,y]\}}{P\{\geq 1 \text{ arrivo in } [0,a]\}} =$$

$$= \frac{P\{0 \text{ arrivi in } [y,a]\} P\{\geq 1 \text{ arrivo in } [0,y]\}}{P\{\geq 1 \text{ arrivo in } [0,a]\}}$$

Poichè il processo degli arrivi è un processo di Poisson, si ottiene

$$= \frac{\mathrm{e}^{-G(a-y)}(1 - \mathrm{e}^{-Gy})}{1 - \mathrm{e}^{-Ga}} = \frac{\mathrm{e}^{-Ga}(\mathrm{e}^{Gy} - 1)}{1 - \mathrm{e}^{-Ga}}$$

Dalla funzione di distribuzione cumulativa si ottiene la media

$$E[b] = \frac{Ga - 1 + \mathrm{e}^{-Ga}}{G(1 - \mathrm{e}^{-Ga})}$$

da cui

$$E[B_c] = \frac{Ga - 1 + \mathrm{e}^{-Ga}}{G(1 - \mathrm{e}^{-Ga})} + 1 + a$$

Esaminiamo ora tre modi diversi per il calcolo del throughput del protocollo NP-CSMA.

Calcolo diretto basato sullo schema funzionale

Il throughput S è ricavabile come $S = GP_LP_S$, dove G è il traffico offerto alla rete, P_L è la probabilità di sentire il canale libero è P_S è la probabilità di avere una trasmissione con successo condizionata dall'avere il canale libero. Lo schema funzionale su cui si basa il calcolo è riportato nella figura 5.12.

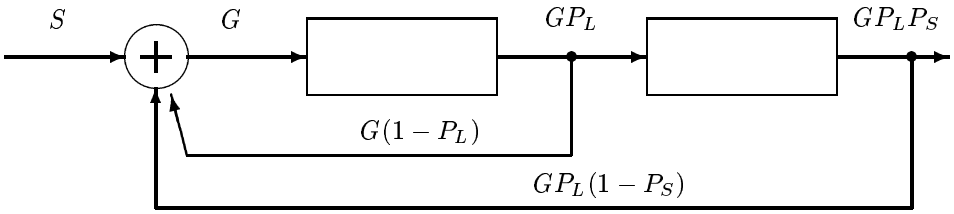


Figura 5.12. Schema funzionale.

Per avere una trasmissione con successo è necessario che la trasmissione sia la prima dopo la fine di un periodo di occupato (altrimenti sicuramente si verifica una collisione tra la trasmissione considerata e la prima) e che nessun'altra stazione trasmetta per un intervallo di tempo a a partire dall'istante di inizio della trasmissione.

Il secondo evento si verifica con probabilità e^{-aG}.

Per il calcolo della probabilità del primo evento si deve considerare che l'inizio della trasmissione deve avvenire nel periodo I del tempo di durata $I + a$ in cui il canale viene sentito libero. Ovvero la trasmissione deve iniziare nell'intervallo in cui tutto il canale è effettivamente libero e non vi possono essere collisioni, come è chiaro dalla figura 5.11.

Ne consegue che

$$P_S = \mathrm{e}^{-aG}\frac{E[I]}{E[I]+a}$$

La probabilità P_L di sentire il canale libero è $1-P_O$, dove P_O è la probabilità di sentire il canale occupato. L'intervallo di tempo in cui il canale è sentito occupato è l'intervallo B tranne che per la sua parte iniziale di durata a in cui il segnale è già in propagazione, ma non può ancora essere sentito. Ne consegue che

$$P_L = 1 - \frac{E[B]-a}{E[I]+E[B]}$$

Allora, poiché

$$E[B] = E[B_C](1-\mathrm{e}^{-aG}) + E[B_S]\mathrm{e}^{-aG}$$

ne consegue che

$$S = \frac{G\mathrm{e}^{-aG}}{\mathrm{e}^{-aG}+G(1+2a)}$$

Analisi di ciclo – primo caso

Il throughput S può essere calcolato facendo riferimento al tempo per cui il canale è utilizzato per una trasmissione con successo all'interno di un ciclo che comprende un periodo libero ed un periodo occupato. Il throughput si può quindi ricavare come rapporto tra il prodotto del tempo di trasmissione di una PDU (1) per la probabilità di avere un periodo di tipo B_S (e^{-aG}) e la durata media di un ciclo, che si può scrivere $E[B] + E[I]$. Si ottiene

$$S = \frac{\mathrm{e}^{-aG}}{E[I] + E[B]} = \frac{G\mathrm{e}^{-aG}}{\mathrm{e}^{-aG} + G(1+2a)}$$

Analisi di ciclo – secondo caso

Consideriamo un secondo tipo di ciclo che comprende una o più collisioni e una sola trasmissione con successo. Un esempio di un ciclo di questo genere è mostrato nella figura 5.13,

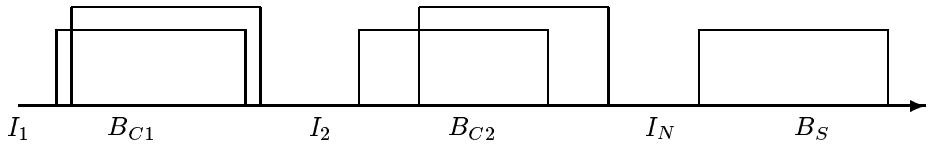


Figura 5.13. Ciclo comprendente una sola trasmissione con successo.

Con una definizione di ciclo di questo tipo, il throughput S può essere calcolato come rapporto tra la durata dell'unica trasmissione andata a buon fine (1) e il valor medio della durata totale del ciclo. Il ciclo è composto da intervalli in cui il canale è libero, da altri in cui si verificano collisioni e da una unica trasmissione andata a buon fine. Sia N la variabile casuale che indica il numero totale di collisioni nel ciclo. Allora

$$S = \frac{1}{E[I]\{E[N]+1\} + E[N]E[B_C] + (1+a)}$$

Bisogna quindi calcolare $E[N]$. Poiché N è una variabile casuale con densità di probabilità geometrica

$$P\{N = k\} = (1 - \mathrm{e}^{-aG})^{k-1}\mathrm{e}^{-aG}$$

si ha

$$E[N] = \mathrm{e}^{aG} - 1$$

Sostituendo si ottiene il risultato in funzione di a e G.

Con un procedimento analogo a quest'ultimo si riesce a calcolare il throughput S anche per la versione 1-persistente del protocollo. Il calcolo è un po' più complicato poiché le PDU non trasmesse immediatamente non vengono riassorbite dal processo degli arrivi a parametro G come nel caso semplificato esaminato finora.

Si ottiene

$$S = \frac{G[1 + G + aG(1 + G + aG/2)]\mathrm{e}^{-G(1+2a)}}{G(1+2a) - (1 - \mathrm{e}^{-aG}) + (1+aG)\mathrm{e}^{-G(1+a)}}$$

Nelle figure 5.14 e 5.15 sono riportati gli andamenti del throughput S al variare di G con a come parametro, per le versioni non persistente e 1-persistente del protocollo. Per il caso non persistente con $a = 0$

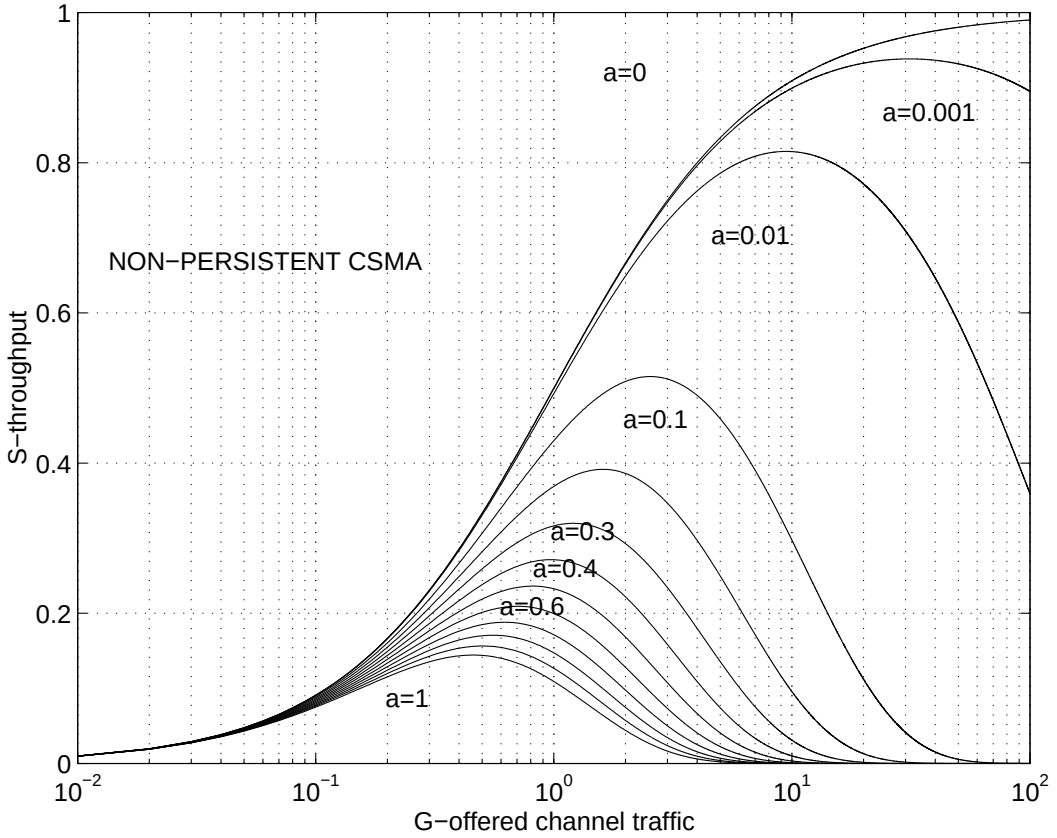


Figura 5.14. Andamento del throughput al variare del traffico offerto per la versione non persistente del protocollo CSMA.

il throughput cresce monotonicamente al crescere del carico G; si arriva ad ottenere $S = 1$ al limite per G che tende ad infinito. Invece, per $a = 0.01$ si ha un throughput massimo circa uguale a $S = 0.8$ per circa $G = 10$. Nel caso della versione 1-persistente per $a = 0$ il massimo throughput è circa 0.6 e la curva non è monotona crescente; in altre parole il protocollo presenta una instabilità anche in questo caso. La causa dell'instabilità è legata alla presenza di collisioni anche con ritardo di propagazione nullo, per il fatto che se il canale viene trovato occupato da più stazioni durante una trasmissione (che può già essere una collisione), quando tale trasmissione termina si verifica una collisione, durante la quale si accoderanno altre stazioni in attesa di trasmettere e così via.

Nella figura 5.16 è riportato l'andamento del throughput in funzione del traffico offerto per le diverse versioni del protocollo CSMA. La zona di interesse delle curve è quella per G compreso tra 0.1 e 1, in cui il protocollo 1-persistente permette di ottenere un throughput più elevato rispetto ai protocolli p-persistenti o alla versione non persistente. Nonostante l'esistenza del fenomeno dell'instabiltà, poiché nella zona a carico G ragionevolmente basso il protocollo 1-persistente ha un throughput superiore, le implementazioni dei protocolli CSMA utilizzano spesso il protocollo CSMA nella versione 1-persistente.

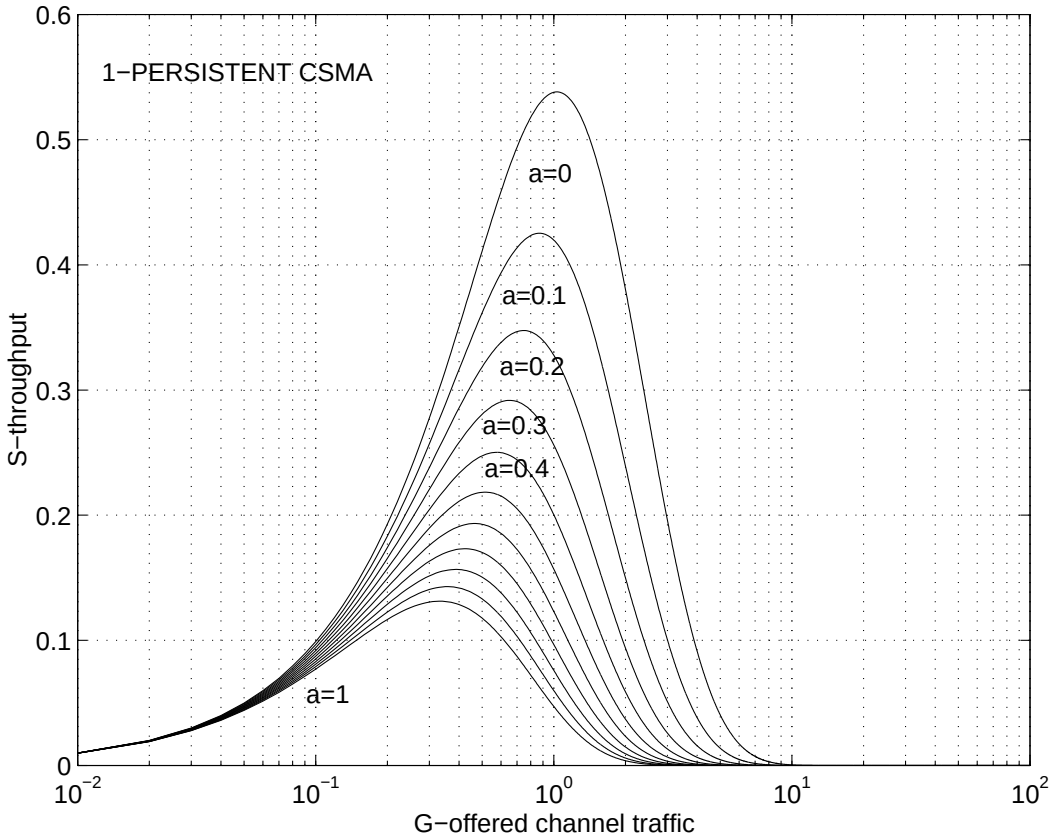


Figura 5.15. Andamento del throughput al variare del traffico offerto per la versione 1-persistente del protocollo CSMA.

5.6.2 I protocolli CSMA/CD

Prestazioni migliori di quelle fornite dai protocolli di tipo CSMA si ottengono se le stazioni hanno la possibilità di riconoscere le collisioni, sospendendo le trasmissioni che occupano inutilmente la risorsa trasmissiva. Si parla in questo caso di protocolli CSMA/CD (CSMA con Collision Detection). Lo standard Ethernet, il più utilizzato nelle LAN, prevede l'utilizzazione di un protocollo CSMA/CD 1-persistente su di una topologia a bus.

Come detto, la caratteristica dei protocolli con Collision Detection è di saper riconoscere le collisioni mentre queste si verificano; le stazioni coinvolte non terminano la trasmissione della PDU (che andrebbe persa) ma si bloccano quasi immediatamente. Per ottenere questo risultato si ascolta il canale non solo prima di trasmettere, ma anche durante la fase di trasmissione in modo da accorgersi il più in fretta possibile dell'avvenuta collisione; visto che il ritardo di propagazione è piccolo rispetto al tempo di trasmissione di una PDU, si ottiene un miglioramento in termini di prestazioni.

L'algoritmo di accesso su cui si basano i protocolli CSMA/CD nella loro versione non persistente è il seguente:

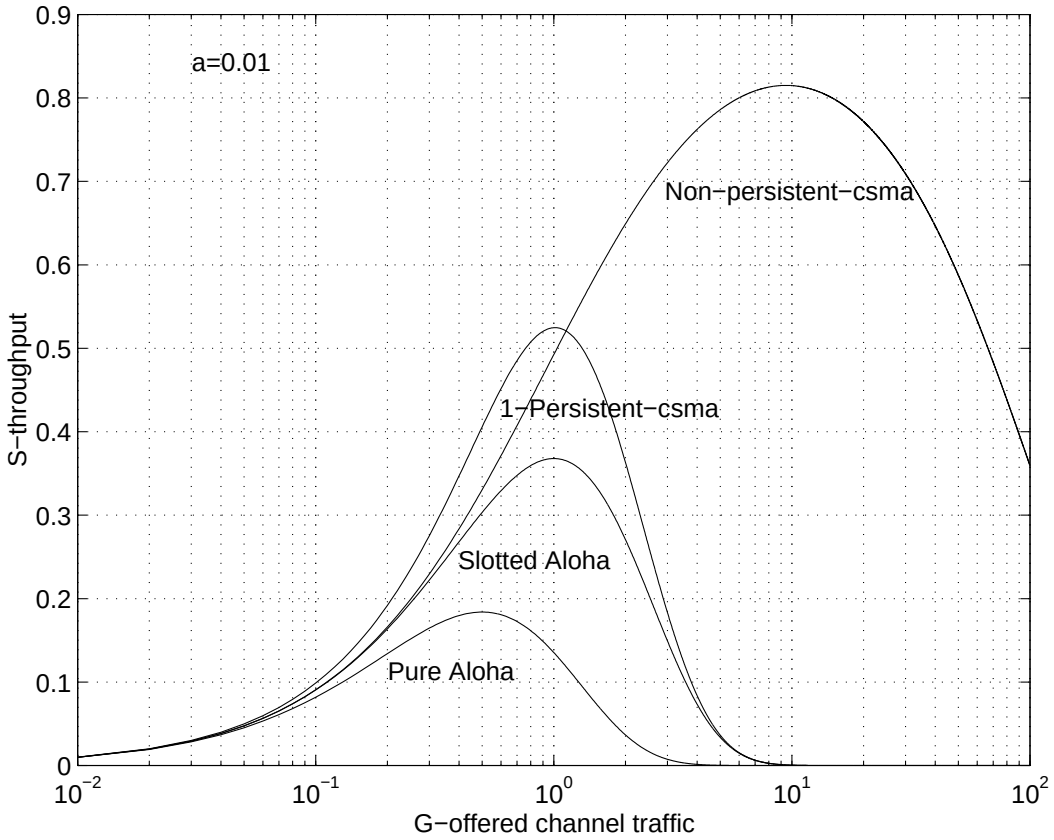


Figura 5.16. Confronto di prestazioni tra le diverse versioni del protocollo CSMA.

- si ascolta il canale prima di trasmettere;

  se il canale è sentito libero si inizia a trasmettere;

  se il canale è sentito occupato si riprova dopo un tempo casuale;

- mentre è in corso la trasmissione di una PDU, si ascolta il canale per verificare che la trasmissione vada a buon fine;

- se ci si accorge di una collisione, si continua a trasmettere per un breve periodo passando dalla PDU ad una sequenza standard (detta di jamming) e poi si sospende la trasmissione. Il tentativo di trasmissione verrà ripetuto dopo un ritardo casuale.

Nella figura 5.17 è riportato un diagramma spazio-temporale che si riferisce ad una trasmissione con collisione; le stazioni A e B e tutte quelle tra loro comprese si accorgono della collisione poiché osservano la sovrapposizione delle PDU trasmesse da A e B. Le stazioni a sinistra (destra) di A (B) osservano la sequenza dei segnali trasmessi da A e da B e non si accorgerebbero dell'avvenuta collisione; per evitare il rischio che queste stazioni interpretino frammenti di PDU come PDU corrette, si forzano tutte le stazioni

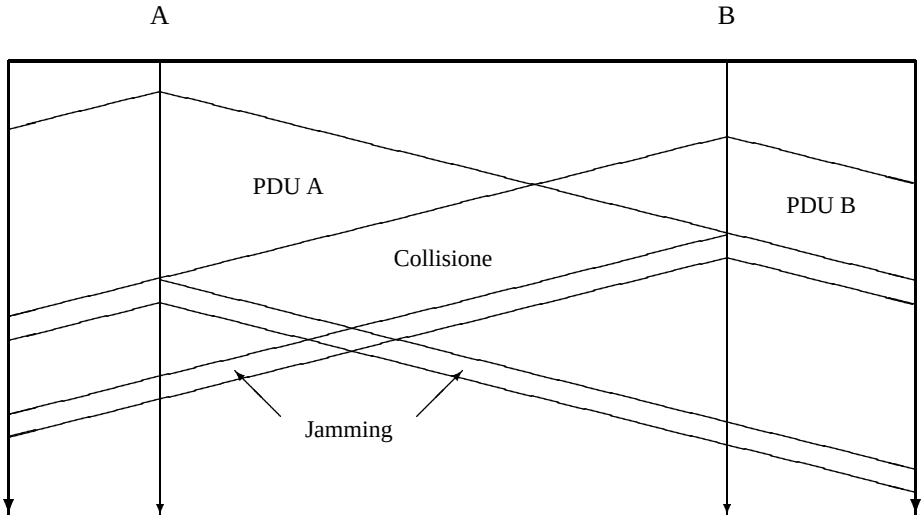


Figura 5.17. Diagramma spazio-temporale della trasmissione di PDU con collisione e sequenza di jamming.

coinvolte nella collisione a trasmettere la sequenza di jamming. Tale sequenza è definita dal protocollo ed il tempo di jamming è descritto da una variabile indicata in unità normalizzate con la lettera σ.

Anche per il calcolo del throughput di questo protocollo si dovrebbe operare sul diagramma spazio-temporale, ma poichè tale approccio risulta complesso, si preferisce svolgere un'analisi approssimata utilizzando solo un asse temporale.

L'intervallo di tempo B_C risulta in questo caso ridefinito come mostrato nella figura 5.18. Sia la stazione 1 che la stazione 2 si accorgono della collisione e trasmettono la sequenza di jamming. Dopo che l'ultima stazione coinvolta nella collisione ha finito di trasmettere la sequenza di jamming, ci sarà ancora un intervallo di tempo a di propagazione in cui il canale è occupato da una trasmissione. Si può ancora osservare nella figura 5.18 che, a causa del ritardo di propagazione, la stazione che ha iniziato per prima a trasmettere è l'ultima ad accorgersi dell'avvenuta collisione.

L'intervallo di tempo σ è composto dal tempo necessario ad una stazione per accorgersi della collisione e dalla durata della sequenza di jamming.

Con le ipotesi fatte, il valor medio $E[B_C] = E[b] + a + \sigma + a$.

Con un procedimento analogo a quello visto in precedenza si ottiene per il throughput S la seguente

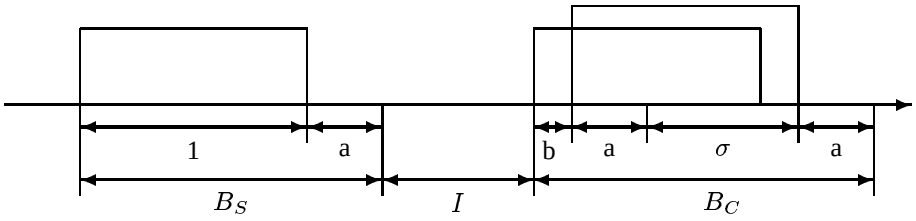


Figura 5.18. Diagramma temporale con sequenza di jamming.

espressione in funzione di a, G e σ:

$$S = \frac{G\mathrm{e}^{-aG}}{[1 + G(1 - a - \sigma)]\mathrm{e}^{-aG} + G(3a + \sigma)}$$

Le prestazioni del protocollo CSMA/CD sono tanto migliori di quelle del protocollo CSMA, quanto più il ritardo di propagazione è basso.

Il protocollo CSMA/CD è molto utilizzato nel caso di LAN, ma non è utilizzabile in reti che utilizzano canali radio, perché in esse non è possibile ascoltare il canale mentre si trasmette poiché, a causa del disadattamento dell'antenna, il ricevitore durante la trasmissione non può sentire altro che il segnale trasmesso.

La versione 1-persistente del protocollo CSMA/CD introduce la stessa variante discussa per il caso CSMA e ha prestazioni migliori di quella non persistente per valori di $G < 1$. Come nel caso dei protocolli CSMA, se in media si tenta di trasmettere meno di una PDU quando il canale è occupato, il protocollo 1-persistente è più conveniente perché si trasmette non appena il canale è libero; se invece, in media più di una PDU tenta la trasmissione per ogni periodo in cui il canale è occupato, la versione non persistente ha prestazioni migliori.

Si osservi che il traffico offerto G non dovrebbe mai avvicinarsi nè tantomeno superare il valore 1 in una rete ben dimensionata, poiché questo significherebbe che si ha un traffico totale (contando anche i tentativi di ritrasmissione) pari o addirittura superiore alla capacità del canale. Il fenomeno dell'instabilità quindi in pratica non si verifica mai, poiché esso si potrebbe presentare solo per valori di traffico offerto molto alti ($G > 10$).

5.7 I protocolli control token

I protocolli ad accesso ordinato comprendono quei protocolli di accesso multiplo nei quali l'assegnazione della risorsa trasmissiva ad ogni utente avviene in modo deterministico ed è regolata da un algoritmo che può essere gestito in modo centralizzato da una unità apposita posta ad un livello gerarchicamente superiore rispetto alle normali stazioni della rete, oppure in modo distribuito, direttamente dagli utenti.

Il primo caso è quello dei protocolli polling, il cui nome deriva dalle realizzazioni che prevedono una stazione, detta Master, che interroga ciclicamente le altre stazioni, dette Slave, abilitandone a turno la trasmissione sul canale. L'operazione di interrogazione ciclica delle singole stazioni da parte del Master viene detta polling.

Il secondo caso è quello dei protocolli chiamati control token, il cui nome deriva dal segnale di controllo che abilita le stazioni a trasmettere, e che viene detto "token".

Poiché in ogni istante di tempo c'è una sola stazione nella rete abilitata a trasmettere, non vi sono ambiguità sull'assegnazione del canale e quindi non si verificano collisioni, a meno di malfunzionamenti.

Nel caso di protocolli MAC per reti che usano canali broadcast si usano protocolli del secondo tipo, con una gestione distribuita dell'algoritmo di allocazione della risorsa. In particolare, un utente può trasmettere solo quando è in possesso del token, che è unico all'interno della rete. Quando la stazione ha finito di trasmettere deve passare il token ad un'altra stazione prestabilita. In questo modo tutte le stazioni della rete entrano ciclicamente in possesso del token, quindi tutti gli utenti possono prima o poi trasmettere.

Con topologie di rete ad anello, è naturale scegliere la sequenza logica secondo cui viene passato il token uguale all'ordinamento fisico delle stazioni. Con topologie di rete che non implicano un ordinamento fisico preciso, come i bus, si può stabilire un ordine logico qualsiasi.

Faremo per il momento riferimento al caso di topologie ad anello. In questo caso si pone il problema dell'estrazione delle PDU dall'anello; infatti, mentre nella topologia a bus le PDU arrivate alla fine del bus sono assorbite dalle terminazioni adattate e le stazioni sono collegate al bus in modo passivo, nelle topologie ad anello i collegamenti al supporto fisico sono attivi; in ogni nodo ci sono un trasmettitore ed un ricevitore e

le PDU sono ricevute, elaborate e ritrasmesse. È necessario eliminare le PDU dall'anello; abbiamo già visto come il protocollo control token regola l'accesso al canale, ma restano da stabilire le regole per l'estrazione delle PDU dall'anello.

In generale, le PDU possono essere estratte:

- dalla stazione destinataria;
- dalla stazione trasmittente;
- da una stazione specializzata nella rete.

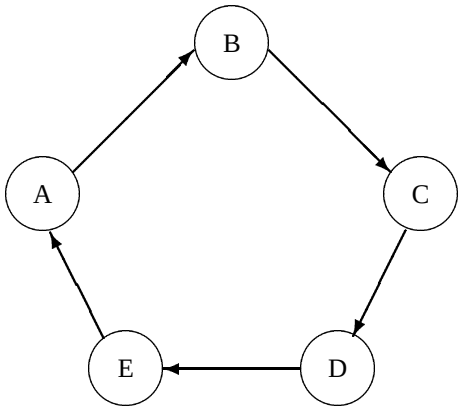


Figura 5.19. Rete ad anello.

L'approccio più naturale è quello di estrarre le PDU alla stazione destinataria. Scegliendo questa opzione il percorso della PDU è il minimo indispensabile, quindi si utilizza la minima quantità possibile delle risorse di rete.

Con riferimento alla disposizione delle stazioni sull'anello nella figura 5.19, esaminiamo una sequenza di trasmissione di PDU.

- Supponiamo che nell'istante iniziale che consideriamo il token arrivi alla stazione A
- A preleva il token dall'anello ed inizia a trasmettere, e dopo avere trasmesso tutta la sua PDU, che supponiamo destinata alla stazione C, ritrasmette il token;
- la PDU transita attraverso la stazione B, che, dovendo trasmettere a sua volta, lascia passare la PDU e preleva il token; trasmette la sua PDU e quindi rilascia il token;
- la stazione C riceve la PDU trasmessa da A ed a lei diretta e la estrae dall'anello; invece la PDU trasmessa da B ed il token vengono ritrasmessi a valle (supponiamo che C non abbia PDU da trasmettere).

Affinchè l'estrazione della PDU da parte del destinatario sia possibile, la stazione C deve identificare nella PDU il suo indirizzo di destinazione prima che la PDU sia passata oltre. È quindi necessario che la stazione C possieda un buffer di transito, inserito tra il canale entrante e il canale uscente, nel quale immagazzinare temporaneamente la PDU per riconoscere a chi è destinata prima di inoltrarla. Il buffer deve essere il più

piccolo possibile per non rendere il ritardo equivalente di propagazione sull'anello troppo grande, poiché le PDU devono attraversare un buffer in ogni nodo dell'anello.

La necessità di avere un buffer in ogni stazione della rete provoca quasi sempre ritardi di propagazione inaccettabili, con conseguente degrado delle prestazioni della rete. Per questo motivo questa soluzione non è sempre la preferita.

Nel caso di estrazione della PDU da parte della stazione sorgente, esistono due possibili modi di funzionamento, detti *single packet* e *single token*. Esaminiamo il caso di funzionamento single packet, cioè a PDU singola.

In questo caso tutte le stazioni hanno un buffer di transito di dimensioni ridottissime, in genere di qualche bit, quindi si tratta di buffer di dimensioni trascurabili rispetto a quelle necessarie per l'estrazione della PDU alla stazione ricevente. In ogni stazione tutte le informazioni che transitano sul canale vengono copiate in un buffer interno, e successivamente analizzate per capire se si tratta di dati destinati alla stazione che devono quindi essere trattenuti, oppure se si tratta di dati destinati ad altre stazioni che possono quindi essere scartati.

Analizziamo il funzionamento single packet facendo riferimento alla figura 5.19. Supponiamo che la stazione A voglia trasmettere una PDU alla stazione C. Quando il token arriva alla stazione A, questa lo preleva dall'anello e trasmette la propria PDU, senza poi rilasciare il token. Tutte le stazioni leggono e ritrasmettono tutti i bit della PDU. La PDU, dopo avere percorso un giro intero sull'anello, torna alla stazione A che la aveva trasmessa e che la estrae dalla rete e solo in seguito all'estrazione rilascia il token. La decisione relativamente all'estrazione è banale, poichè sulla rete circola sempre una sola PDU, quindi non esistono ambiguità. Nella rete c'è al massimo una PDU, trasmessa dalla stazione che ha il token in quell'istante.

Il funzionamento single packet presenta alcuni svantaggi. Ogni PDU compie un intero giro della rete, mentre mediamente, nel caso di traffico uniforme, solamente mezzo giro sarebbe necessario per raggiungere la destinazione. Ciò significa che si usa il doppio delle risorse di rete necessarie. Inoltre, il token non viene rilasciato immediatamente alla fine della trasmissione della PDU. Ciò significa che il canale rimane inutilizzato tra la fine della trasmissione di una PDU e l'inizio della ricezione della PDU stessa da parte della stazione che lo ha trasmesso.

Si ricordi che il vantaggio relativo al funzionamento single packet è quello di richedere in ogni stazione un buffer di ridottissime dimensioni e quindi un ritardo di transito di pochi bit.

Lo svantaggio della scarsa utilizzazione del canale dovuto al ritardato rilascio del token è eliminato dal protocollo single token. Vediamo la sequenza di funzionamento in questo caso. Il token raggiunge la stazione A che preleva il token e inizia la trasmissione di una PDU verso la stazione C. Il token è rilasciato immediatamente alla fine della trasmissione della PDU, senza aspettare di estrarre la PDU dall'anello. La stazione B, che sta copiando la PDU perché non sa a chi è destinata, alla fine della PDU preleva il token, inizia la trasmissione della sua PDU accodandola alla PDU trasmessa dalla stazione A e alla fine della trasmissione rilascia immediatamente il token. Sia A sia B sanno di dovere estrarre la propria PDU e, se non ci sono malfunzionamenti, poiché le PDU non possono superarsi sull'anello, ogni stazione, estraendo la prima che riceve, toglierà la PDU da lei trasmessa.

Questo protocollo è più efficiente del precedente, ma in caso di errore da parte di una stazione rimane una PDU vagante nell'anello, che è molto difficile riuscire ad identificare e che occupa inutilmente una parte delle risorse di rete. È necessario un gestore centralizzato che, ad esempio, segni tutte le PDU con un flag che indica il passaggio attraverso il nodo in modo da riconoscere le PDU che hanno compiuto più di un giro sull'anello. La stazione addetta a questo compito, per poter estrarre dall'anello le PDU che hanno compiuto più di un giro, deve essere dotata di un buffer di transito di dimensioni non trascurabili.

5.7.1 Prestazioni dei protocolli control token

In un protocollo control token (o polling) il permesso di trasmettere circola tra le diverse stazioni. Nella costruzione di un modello per la valutazione delle prestazioni, ognuna delle N stazioni può essere rappresentata con una fila di attesa a cui i clienti (che rappresentano le PDU) arrivano secondo un processo di Poisson a parametro λ, mentre il passaggio del token e le trasmissioni sono rappresentate da un servitore che ciclicamente serve le singole code. Il servitore serve una stazione quando c'è almeno una PDU in coda. Il tempo di servizio è il tempo di trasmissione di una PDU.

Il problema in questo modello è il tempo di spostamento non nullo del servitore da una coda alla coda successiva. Se il tempo di spostamento fosse nullo, il modello sarebbe equivalente, dal punto di vista del ritardo medio di una PDU, ad un'unica coda con processo degli arrivi a parametro $N\lambda$.

Il ritardo di spostamento non nullo del servitore porta ai modelli chiamati multicoda o polling.

Il modello polling che studiamo è composto da N file di attesa a capacità infinita, con processi di arrivo Poisson con parametro λ, tempo di spostamento del servitore da una coda alla successiva costante e pari ad r e tempo di servizio costante e pari a τ.

Calcoliamo inizialmente il valore medio del tempo di rotazione del servitore R, ovvero il tempo tra due arrivi consecutivi del servitore ad una coda qualsiasi. R ha due componenti: il tempo di spostamento lungo il ciclo che è pari ad Nr e la somma dei tempi di servizio lungo il ciclo, che è pari ad $M\tau$ se M è il numero di code a cui viene fornito servizio; il numero di servizi in un ciclo non è costante, ma è una variabile casuale con densità di probabilità non nota. Quindi $R = Nr + M\tau$, ed $E[R] = Nr + E[M]\tau$.

Calcoliamo ora il traffico ρ alla singola coda. Il traffico è pari al prodoto della velocità di arrivo per il tempo medio di servizio equivalente; poiché il tempo medio di servizio è il tempo dall'inizio del servizio all'inizio del servizio successivo, $\rho = \lambda E[R]$.

La probabilità che una coda sia non vuota è ρ. La probabilità di trovare k code non vuote su N **non** è uguale a $\binom{N}{k}\rho^k(1-\rho)^{N-k}$, poiché non si ha indipendenza statistica tra le diverse code. Se una coda ha molti clienti in attesa, la probabilità di avere molti clienti in attesa anche alle altre code è alta, perché il servitore si ferma a servire la coda molto carica.

Introduciamo lo stesso l'ipotesi di indipendenza statistica che semplifica grandemente i conti. Possiamo allora calcolare $E[M] = N\rho$ e $\sigma_M^2 = N\rho(1-\rho)$ da cui si può ricavare $E[R] = Nr+N\rho\tau = Nr+N\lambda\tau E[R]$. Risolvendo per $E[R]$ si ottiene

$$E[R] = \frac{Nr}{1 - N\lambda\tau}$$

Inoltre sappiamo che

$$\rho = \frac{Nr\lambda}{1 - N\lambda\tau}$$

e

$$E[M] = N\rho = \frac{N^2 r\lambda}{1 - N\lambda\tau}$$

Poiché vale la relazione $R = Nr + M\tau$ si possono calcolare tutti i momenti di R a partire da quelli di M.

Calcoliamo ora il ritardo medio di una PDU. Nella figura 5.20 gli istanti di arrivo del servitore alla coda sono indicati con una freccia verticale. Supponiamo che all'origine dei tempi arrivi la PDU E da trasmettere e ce ne siano già quattro in coda. Si vuole calcolare il ritardo medio della PDU, ovvero il tempo tra l'istante di arrivo della PDU alla coda e l'istante in cui l'ultimo bit della PDU lascia la stazione. Dalla figura si può vedere che

$$E[T] = \tau + E[R]E[Q] + E[R_r]$$

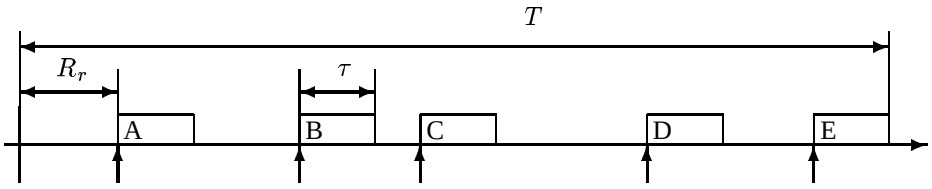


Figura 5.20. Diagramma temporale per l'analisi di protocolli control token

dove Q è il numero di PDU nella fila di attesa quando arriva la PDU E e R_r è il tempo residuo di rotazione del servitore a partire dall'istante di arrivo di E.

Il tempo medio di attesa nella fila è $E[W] = E[T] - \tau = E[R]E[Q] + E[R_r]$. Per il teorema di Little $E[Q] = \lambda E[W] = \lambda E[R]E[Q] + \lambda E[R_r]$, da cui

$$E[Q] = \frac{\lambda E[R_r]}{1 - \lambda E[R]}$$

Dobbiamo ancora calcolare $E[R_r]$; il tempo di rotazione non ha densità di probabilità esponenziale, quindi si deve utilizzare la relazione generale

$$E[R_r] = \frac{E[R^2]}{2E[R]} = \frac{1 + C_R^2}{2} E[R]$$

dove C_R è il coefficiente di variazione del tempo di rotazione del servitore. Si noti che nel caso deterministico tale relazione si riduce a $E[R_r] = E[R]/2$, mentre nel caso esponenziale, si ricava $E[R_r] = E[R]$, coerentemente con la assenza di memoria.

Sostituendo si ottiene allora

$$E[T] = \frac{(1 + C_R^2)E[R]}{2(1 - \lambda E[R])} + \tau$$

Nella figura 5.21 sono riportati gli andamenti del ritardo per i protocolli CSMA/CD e token ring con una velocità di trasmissione di 10Mbit/s; la rete è composta da cinquanta stazioni e per il protocollo token le stazioni hanno un solo bit di ritardo di inserzione. Non si considerino le curve relative al protocollo slotted ring e al protocollo MLMA, un protocollo a token implicito su struttura a bus. Si noti come a 10 Mbit/s il protocollo token ring si comporta in modo significativamente migliore. La latenza di un bit è però un parametro critico e, se non si riesce ad ottenerlo, le prestazioni possono peggiorare in modo anche significativo.

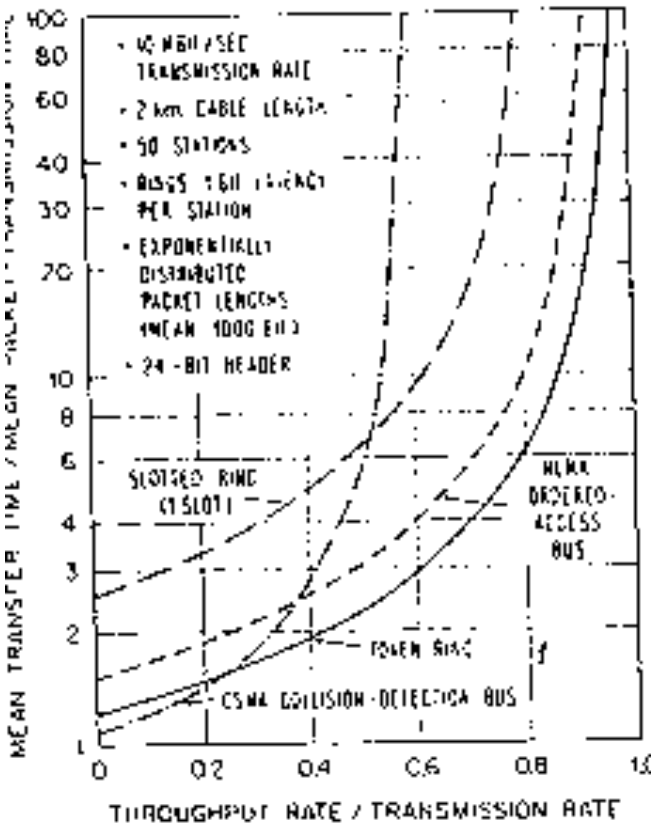


Figura 5.21. Ritardo normalizzato in reti a 10Mbps.

5.8 Gli standard IEEE 802 per le LAN

Il comitato IEEE 802 è stato costituito per la standardizzazione di reti locali (LAN) e la sua attività ha prodotto tutta una serie di standard, tra cui:

- IEEE 802.3 o CSMA/CD, derivato da Ethernet;
- IEEE 802.4 o token bus;
- IEEE 802.5 o token ring.

Lo standard Ethernet è basato su una topologia a bus con un protocollo CSMA/CD 1-persistente ed è nato per l'ambiente di automazione di ufficio. Lo standard token bus è basato su una topologia a bus con un protocollo control token ed è nato per un ambiente di automazione di fabbrica. Lo standard token ring è basato su una topologia ad anello con un protocollo control token e cerca di risolvere i problemi di entrambi gli ambienti.

I ritardi nel trasferimento delle PDU, per evidenti motivi, possono essere critici in ambiente di automazione industriale, mentre non lo sono in applicazioni di ufficio. Il fatto che il protocollo control token permetta di definire un ritardo di accesso massimo alla rete in funzione del tempo di rotazione del token, mentre ciò non è possibile per il protocollo CSMA, spiega il motivo del successo ottenuto in ambiente di automazione industriale dai protocolli control token.

Dal punto di vista delle topologie, un anello è meno affidabile di un bus (a causa delle inserzioni attive delle stazioni sul canale). Inoltre, la probabilità di guasto di un anello cresce con il numero di stazioni collegate sull'anello: se p è la probabilità di guasto di una stazione, la probabilità che un anello di N stazioni funzioni correttamente vale $(1-p)^N$.

5.8.1 Lo standard Ethernet

Lo standard Ethernet è nato dalla collaborazione tra Digital, Intel e Xerox e riguarda il livello fisico e il livello collegamento del modello OSI. Sono state pubblicate due versioni (Ethernet v.1 e Ethernet v.2) della raccomandazione Ethernet. Da Ethernet v.2 è nato lo standard IEEE 802.3, che si differenzia solo per alcuni dettagli, e che è con esso compatibile. In ambito IEEE 802.3 ci sono poi state continue evoluzione degli standard, soprattutto per quanto riguarda i mezzi trasmissivi e la velocità di trasmissione. In questo paragrafo si fa principalmente riferimento alla specifica Ethernet v.2.

Per quanto riguarda il livello fisico lo standard Ethernet prevede:

- l'uso di un cavo coassiale (spesso o sottile) come supporto fisico;
- trasmissione in banda base con codifica Manchester;
- topologia ad albero senza radice;
- velocità di trasmissione di 10 Mbit/s;
- fino a 1024 stazioni su una sola rete;
- massima distanza tra due stazioni pari a 2800 metri.

Il protocollo di livello MAC è un CSMA/CD 1-persistente con PDU di dimensione variabile da 64 a 1518 byte.

Livello fisico

Nella figura 5.22 è riportato lo schema di principio di una stazione Ethernet. I suoi componenti sono il controller, che si occupa della gestione vera e propria del protocollo, il transceiver, ovvero il rice-trasmettitore, ed il transceiver cable, che collega il controller al transceiver. Il transceiver è collegato al cavo coassiale con uno spillo, senza interrompere il cavo; lo spillo si comporta come un'iride nella guida d'onda del cavo coassiale.

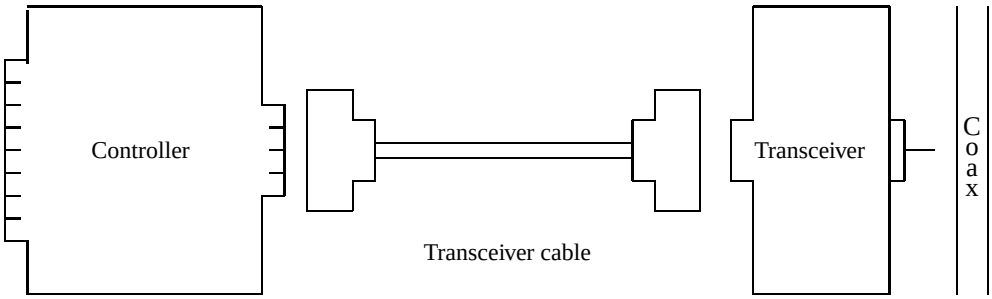


Figura 5.22. Schema di principio di una stazione Ethernet.

Lo schema descritto è quello previsto dall'originale standard Ethernet, che oggi è in parte obsoleto; nelle reti di PC e *workstation* oggi si utilizza come mezzo trasmissivo o un cavo coassiale sottile (RG 58), che viene tagliato in corrispondenza di ogni stazione per permetterne il collegamento con le schede di rete usando connettori BNC, o un doppino che collega direttamente le stazioni a dispositivi di interconnessione. I protocolli d'accesso e di livello fisico sono implementati da uno o più chip contenuti nella scheda di rete e si elimina la necessità del transceiver cable. La lunghezza massima del cavo sottile è di 180 metri e a questo spezzone si possono collegare fino a 30 utenti; La lunghezza consentita sul doppino è invece di 100 m. Il cavo coassiale "spesso" è oggi ancora talvolta utilizzato solo per reti di backbone a cui sono collegate le diverse reti di distribuzione che portano i segnali agli utenti.

La topologia ad albero senza radice è formata da vari segmenti di bus collegati tra loro. Nel caso di cavo coasiale spesso, ogni segmento può essere lungo fino a 500 metri e si possono collegare fino a 100 utenti sullo stesso segmento; poiché la lunghezza massima del transceiver cable è di 50 metri, la distanza massima tra due utenti sullo stesso segmento è di 600 metri. La configurazione può essere aumentata collegando fino a tre segmenti mediante ripetitori, raggiungendo una lunghezza massima di 1500 metri di cavo coassiale. I ripetitori rigenerano i segnali elettrici del livello fisico. Due ripetitori possono essere collegati da un canale punto punto lungo 1000 metri; aggiungendo i 300 metri dei transceiver cable (si hanno infatti 6 tratte di tale tipo di cavo) si raggiunge una distanza massima tra stazioni di 2800 metri. Nelle figure 5.23, 5.24 e 5.25 sono riportate tre configurazioni possibili per una LAN Ethernet. Lo standard IEEE 802.3 prevede vincoli diversi per il numero di ripetitori e di segmenti che possono essere concatenati: per esempio, con il cavo coassiale sottile, si possono avere fino a cinque spezzoni di cavo in serie (interconnessi quindi da quattro ripetitori).

Per quanto riguarda gli aspetti trasmissivi, lo standard prevede una trasmissione in banda base con codifica Manchester. Questa codifica consiste nell'adottare per i simboli binari 0 e 1, forme d'onda come quelle della figura 5.26, in modo da poter facilmente ricuperare il sincronismo di simbolo anche in presenza di lunghe sequenze di 1 o di 0. Purtroppo con queste forma d'onda la banda richiesta è in prima approssimazione il doppio della velocità di segnalazione.

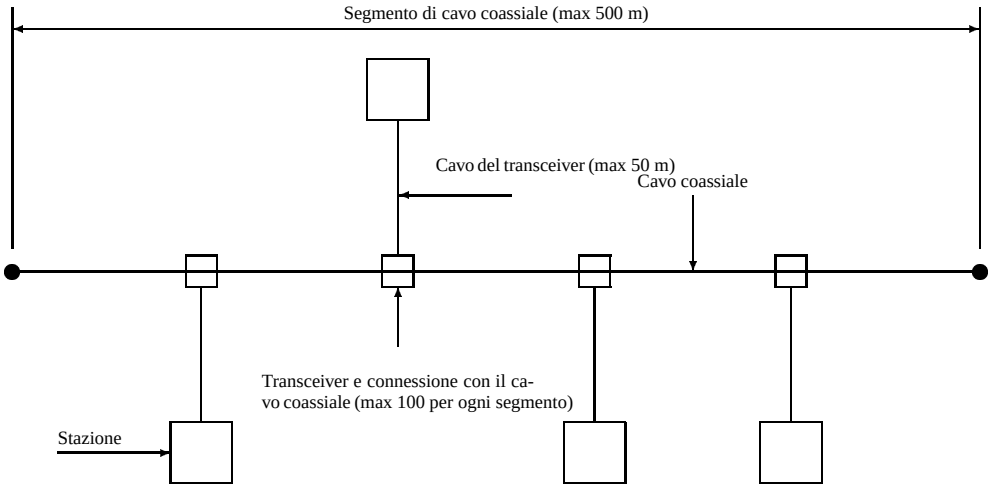


Figura 5.23. Configurazione minima

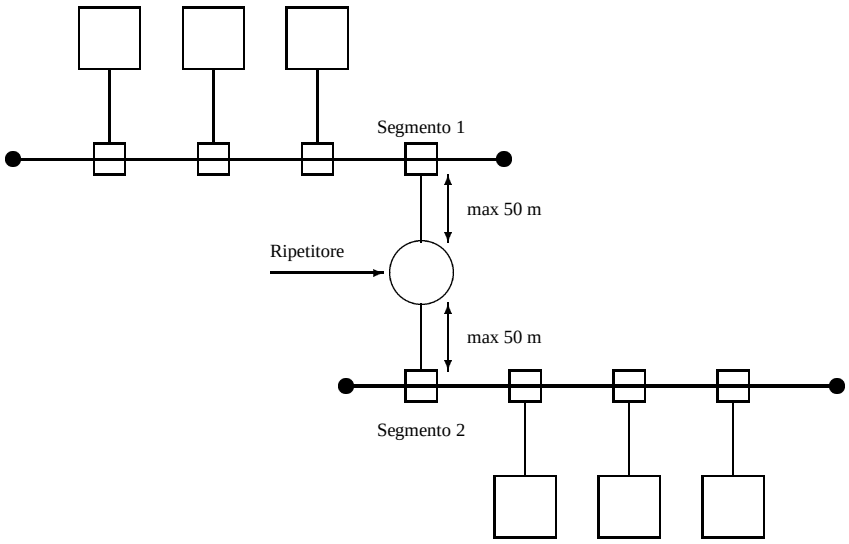


Figura 5.24. Configurazione media

Livello data link

La PDU a livello data link ha il formato riportato nella figura 5.27; gli indirizzi occupano 6 byte, il campo dati è di lunghezza variabile da 46 a 1500 byte e il campo CRC è di 4 byte. Il campo tipo (*type*) occupa 2 byte, e assume significati diversi in Ethernet v.2 e in IEEE 802.3: nel primo caso contiene una codifica del

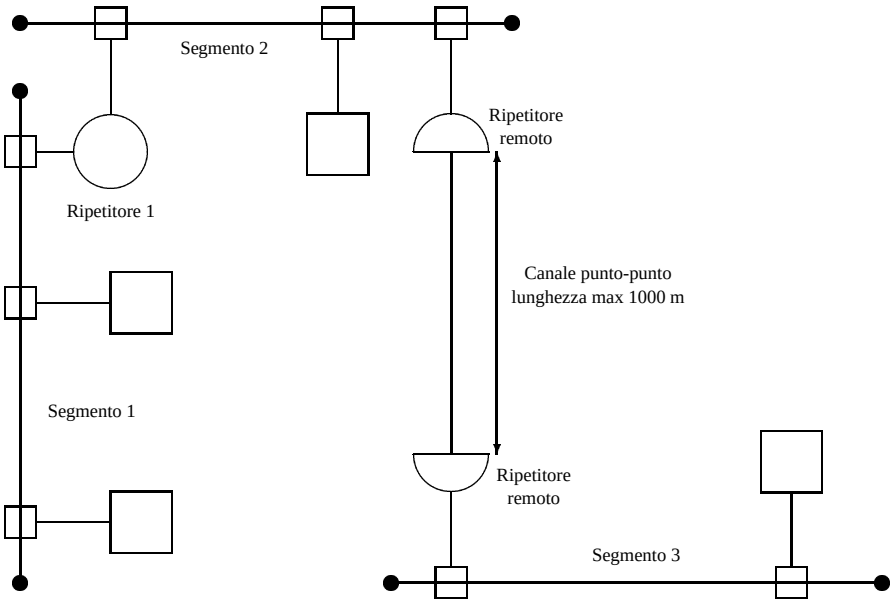


Figura 5.25. Configurazione massima

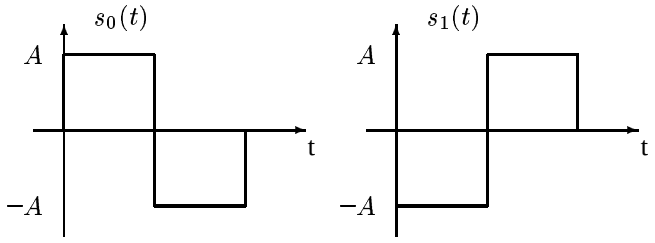


Figura 5.26. Forme d'onda utilizzate per la codifica Manchester

protocollo di livello superiore cui i dati contenuti nel pacchetto sono destinati (per esempio IP è codificato con 0X0800), mentre caso di IEEE 802.3 contiene il numero di byte significativi nel campo dati (si ricordi che il campo dati non può essere inferiore a 46 byte, per cui non tutti i byte in esso contenuto sono necessariamente significativi). Visto che il numero di byte nel campo dati non può essere maggiore di 1500, e tutti i protocolli di livello superiore sono codificati con numeri più grandi di 1500, le due modalità di utilizzo del campo di tipo sono tra di loro compatibili e possono coesistere nella stessa rete Ethernet.

La scelta di un campo dati che può raggiungere dimensioni così ampie è dettata da varie ragioni:

- poiché le prestazioni del protocollo CSMA/CD sono tanto migliori quanto più il parametro a (rapporto tra il tempo di propagazione ed il tempo di trasmissione della PDU) è piccolo, aumentando la dimensione della PDU si riesce a migliorare l'efficienza della rete;
- la durata del periodo di jamming dipende dall'implementazione hardware, ed il suo impatto risulta

Figura 5.27. Formato della PDU a livello data link

tanto minore quanto più è grande la dimensione della PDU.

Si osservi inoltre che vengono utilizzati ben 6 byte per i campi di indirizzo sorgente ed indirizzo destinazione. Questi campi possono apparire di ampiezza esagerata rispetto alle effettive necessità di indirizzamento: infatti con 6 byte si possono indirizzare 2^{48} (circa 3×10^{14}) stazioni, mentre una rete Ethernet non può comprendere più di 1024 stazioni. La ragione di questa scelta è dovuta al fatto che ogni scheda Ethernet (su qualsiasi rete) ha un indirizzo unico e diverso da quello di qualunque altra scheda (sulla stessa rete o su di un'altra rete).

Gli indirizzi utilizzabili effettivamente sono in realtà in numero un po' minore a quanto calcolato prima, in quanto l'indirizzo di tutti 1 è utilizzato per i messaggi broadcast, mentre i messaggi multicast sono inviati utilizzando indirizzi in cui il primo bit è a 1.

Sottolivello MAC

Il protocollo MAC è, come già detto, il CSMA/CD 1-persistente.

Le unità di temporizzazione usate dal protocollo sono il tempo di bit, pari a $0.1\mu s$ e lo slot, pari a 512 tempi di bit. La dimensione dello slot è stata definita in modo da essere superiore al ritardo di propagazione end-to-end (che nella configurazione massima della rete arriva a 450 tempi di bit), al tempo di acquisizione del canale ed alla lunghezza dei frammenti di PDU.

Quando una stazione ascolta il canale e lo trova occupato, aspetta che questo si liberi; quando ciò avviene si aspetta per ulteriori 96 tempi di bit e poi si trasmette.

Durante la trasmissione, se si rivela una collisione, si interrompe la trasmissione della PDU e si trasmette la sequenza di jamming per un tempo di 32 – 48 bit.

Dopo 16 tentativi di trasmissione falliti per collisione si rinuncia e si passa al livello superiore l'informazione di trasmissione non riuscita con una primitiva MA_DATA.confirm.

Finché si sta ritrasmettendo una PDU, non si considerano nuove PDU per la trasmissione.

Il ritardo casuale prima della ritrasmissione di una PDU è espresso in slot, ed è una funzione del numero d'ordine della trasmissione: se n rappresenta il numero d'ordine della prossima trasmissione della PDU, e k è il minimo tra n e 10, il ritardo casuale è pari ad un numero di slot uniformemente distribuito tra 0 e 2^k.

Questo aumento del ritardo di ritrasmissione in funzione di n tiene conto del fatto che se il traffico è basso (e quindi ci sono poche collisioni) conviene cercare di ritrasmettere in fretta, mentre al crescere del traffico sul canale (quando ci sono molte collisioni) è meglio cercare di non sovraccaricare il canale aumentando il tempo di ritrasmissione.

Quando si trasmette una PDU, si fa precedere la trasmissione dei dati utili da un preambolo (sequenza di 64 bit, 0 e 1 alternati) che permette ai ricevitori di acquisire il sincronismo di bit. Tale preambolo funge anche da delimitatore iniziale del pacchetto. La delimitazione finale del pacchetto è ottenuta con un periodo di silenzio detto *Inter-Packet Gap* (IPG), che deve essere almeno pari a 96 tempi di bit (o 12 tempi di byte).

5.8.2 Lo standard token ring

Lo standard token ring prevede una topologia ad anello, con velocità di trasmissione di 2, 4, o 16 Mbit/s utilizzando come mezzo di trasmissione il doppino telefonico o il cavo coassiale o la fibra ottica.

Lo standard IEEE 802.5 definisce il protocollo MAC, il formato delle PDU, e le procedure di rilevazione dei guasti e dei malfunzionamenti.

Per migliorare l'affidabilità della topologia ad anello, lo standard prevede l'uso di concentratori di cablaggio (*wiring concentrator*) come nella figura 5.28; in questo modo è più semplice eliminare dall'anello una stazione guasta. Spesso è utilizzato anche un doppio anello, riconfigurabile come anello semplice in caso di malfunzionamenti lungo i canali tra un nodo e l'altro.

Il protocollo MAC è di tipo single packet. Il ritardo di inserzione in ogni stazione è molto piccolo; le stazioni non hanno il tempo di leggere il token e di prelevarlo dalla rete. Il token viene trasformato in un preambolo di PDU, modificando un bit soltanto; alla fine della trasmissione della PDU, si attende di iniziare l'estrazione della PDU dalla LAN e poi il token viene nuovamente inserito nell'anello.

I formati delle PDU e del token sono riportati nella figura 5.29. La PDU è di dimensione variabile ed è composta da 7 o 15 byte di intestazione, dal campo dati e da 2 o 6 byte di coda. L'intestazione comprende 1 byte di delimitazione (SD, Starting Delimiter), 2 byte di controllo (AC, Access Control, e FC, Frame Control), 2 o 6 byte di indirizzo destinazione (DA, Destination Address), 6 byte di indirizzo sorgente (SA, Souce Address), mentre la coda è composta da 4 byte di CRC (FCS), da 1 byte di stato (FS, Frame Status), che è utilizzato per inviare un ACK immediato e che contiene messaggi quali indirizzo riconosciuto, PDU copiata, e da 1 byte di delimitazione (ED, Ending Delimiter). I 2 byte di controllo dell'intestazione (AC e FC) sono utilizzati per scambiare informazioni sul token, per le prenotazioni e per le indicazioni sul formato.

Il campo ACCESS CONTROL è quello che permette di implementare un protocollo control token con priorità. Esso contiene tre bit (PPP) che definiscono la priorità della PDU, tre bit (RRR) utilizzati per la richiesta di priorità, cioè per indicare la massima priorità delle PDU che una stazione vuole trasmettere e favorire così le PDU a priorità più alta, un bit (M) utilizzato dalla stazione che funge da monitor per identificare le PDU spurie nell'anello e un bit (T) per identificare se la PDU è un token ($T = 0$) o una PDU dati ($T = 1$).

Questa particolare implementazione del protocollo control token prevede che le singole stazioni facciano una richiesta di priorità, cioè comunichino la necessità di spedire PDU con una certa priorità. Il token viene generato con la priorità più alta tra quelle richieste dalle stazioni. Una stazione potrà trasmettere sul canale solo se le sue PDU hanno priorità maggiore o uguale a quella del token, altrimenti dovrà passare il token e aspettarne un altro di priorità sufficientemente bassa.

Gli indirizzi possono essere su 48 o su 16 bit. I formati sono simili: un bit indica se l'indirizzo è universale

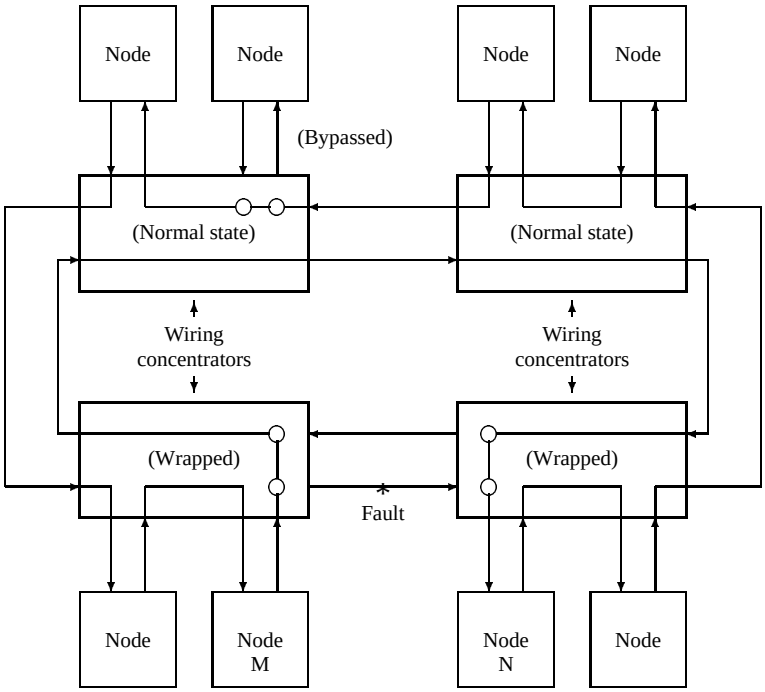


Figura 5.28. LAN Token Ring con wiring concentrator

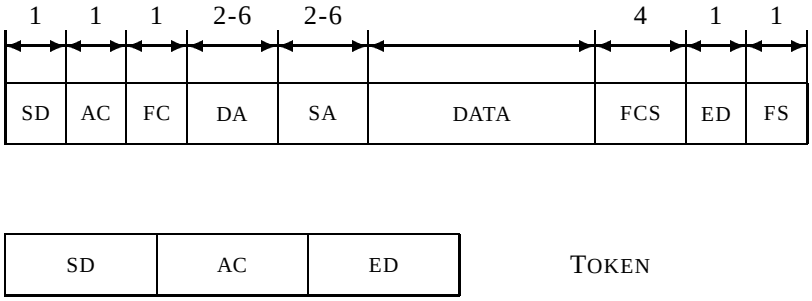


Figura 5.29. Formato di una PDU 802.5

o locale. Se la rete è formata da molti anelli collegati tra loro, l'indirizzo è composto da indirizzo dell'anello (14 bit) e indirizzo dell'utente (32 bit).

Nel byte FRAME STATUS, utilizzato per il meccanismo di ACK immediato, i due bit importanti (sono ripetuti due volte) sono il bit A e il bit C. Se $C = 1$ significa che i dati sono stati copiati, mentre il bit $A = 1$

indica che la stazione è esistente. Questo meccanismo di ACK a livello MAC non è previsto dallo standard OSI, ma è stato realizzato perché utile e molto semplice da implementare.

Nella rete deve esistere una sola stazione che faccia da monitor, eletta all'atto della configurazione iniziale dell'anello. La stazione che immette il token sull'anello assume il ruolo di monitor e lo mantiene fino a quando non si verifica una reinizializzazione. La stazione monitor gestisce il token e si occupa dell'eliminazione delle PDU spurie, utilizzando il bit M del byte ACCESS CONTROL e un buffer interno di 24 bit che garantisce il contenimento del token nell'anello.

Lo standard 802.5 prevede che tutte le stazioni si sincronizzino sulla temporizzazione dei bit del token. Poichè la stazione monitor genera il token, essa definisce quindi anche un sincronismo globale di rete.

5.8.3 Lo standard token bus

Lo standard IEEE 802.4 è nato in ambiente automazione di fabbrica e riguarda come al solito il livello fisico e il livello data link del modello OSI.

Il livello fisico definisce una topologia a bus su cui la trasmissione può avvenire

- o con FSK a fase continua a 1 Mbps con codifica Manchester;
- o con FSK coerente a 5-10 Mbps come per la trasmissione di segnale televisivo via cavo (CATV);
- o con AM/PSK multilivello a 1-10Mbps come per la CATV.

Il protocollo è di tipo control token e il sottolivello MAC descrive le modalità per

- creare e mantenere l'anello logico tra le stazioni;
- gestire il token;
- gestire le PDU dati;
- recuperare gli errori.

Una stazione può accedere al mezzo trasmissivo per un tempo massimo prefissato solamente quando possiede il token. Questo è inviato da una stazione alla stazione che la segue in base all'ordine *logico* stabilito in fase di inizializzazione della rete. Il token ha quindi un indirizzo di sorgente e un indirizzo di destinazione ed è una vera e propria PDU di controllo (a differenza del caso token ring). Poiché non esiste un anello fisico che impone una relazione d'ordine tra le stazioni sulla rete, ogni stazione deve sapere qual è la stazione che la segue nell'ordine logico; per il corretto funzionamento degli algoritmi che vedremo sotto, è anche necessario che ogni stazione sappia quale stazione la precede nell'anello logico. Il passaggio del token richiede maggiore attenzione rispetto al protocollo token ring.

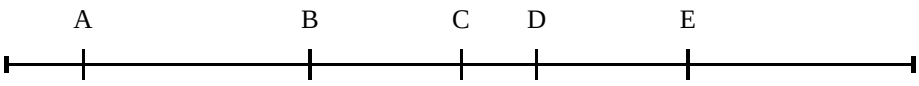


Figura 5.30. Esempio di disposizione delle stazioni in una topologia a bus

Gestione del token

Facciamo riferimento alla figura 5.30 e supponiamo che nella fase di inizializzazione della rete sia stato instaurato tra le stazioni l'ordine logico C, E, D, B, A (che deve essere ciclico, per cui ad A segue C); ogni stazione deve conoscere l'identità della stazione che la segue nell'ordine logico e di quella che la precede.

Esaminiamo il passaggio del token dalla stazione C alla stazione E. C invia il token (una PDU di controllo che contiene E come indirizzo di destinazione) ad E e aspetta di osservare sul bus la trasmissione di una PDU che abbia E come indirizzo di sorgente (E deve o trasmettere una propria PDU dati o passare il token a D) per accertarsi che la stazione E abbia ricevuto il token e lo abbia riconosciuto.

Se la stazione C osserva sul bus una PDU trasmessa da E, sicuramente la gestione del token è passata alla stazione E. Se la stazione C non osserva sul bus una PDU trasmessa da E per un intervallo temporale pari a quattro slot, conclude che l'operazione di passaggio del token non è andata a buon fine e quindi prova una seconda volta, ritrasmettendo il token ad E; dopo due tentativi falliti, si presume che la stazione E non sia funzionante e si rinuncia a passare il token a E.

La rinuncia di C per il guasto di E non può portare al blocco della rete; bisogna attivare una procedura per l'esclusione di E dall'anello logico e passare il token alla stazione successiva ad E. Per identificare la stazione successiva ad E nell'ordinamento logico, la stazione C invia una PDU di controllo di tipo WHO_FOLLOWS$\{E\}$ sulla rete. Ogni stazione, quando osserva una PDU di controllo di tipo WHO_FOLLOWS confronta il parametro della PDU (E nel nostro esempio) con l'indirizzo della stazione che la precede nell'ordine logico. Se i due coincidono (come avviene nel nostro esempio per la stazione D) si risponde alla PDU WHO_FOLLOWS con un'altra PDU di controllo di tipo SET_SUCCESSOR indicando il proprio indirizzo come parametro. Nel nostro caso, la stazione C, vedendo la risposta SET_SUCCESSOR$\{D\}$ può ricostituire la sequenza logica che esclude E, scrivendo l'indirizzo di D al posto di quello di E nella zona di memoria dove si registra l'identificatore della stazione che segue nell'ordine logico.

Se invece nessuno risponde per due volte consecutive all'invio della PDU di controllo di tipo WHO_FOLLOWS$\{E\}$, è necessario iniziare una procedura di reinizializzazione dell'anello inviando una PDU di controllo di tipo SOLICIT_SUCCESSOR.

Tutte le PDU di controllo hanno corrispondenti finestre temporali entro cui si aspetta una risposta; chi trasmette attende sempre una risposta in modo da passare il controllo a qualche altro nodo della rete, per evitare di perdere il controllo del token e dell'ordine logico delle stazioni, così da non dover eseguire una procedura completa di reinizializzazione della rete.

Inserimento di una stazione nell'anello logico

Come esiste un algoritmo per l'esclusione delle stazioni guaste dall'ordine logico, deve esistere anche un algoritmo che permetta a nuove stazioni, o a stazioni che dopo un guasto sono state riparate, di inserirsi nell'anello logico. L'algoritmo è basato sull'invio di PDU di controllo di tipo SOLICIT_SUCCESSOR_1 o SOLICIT_SUCCESSOR_2.

L' algoritmo prevede l'inserzione di una stazione alla volta; se più di una stazione tenta di inserirsi nell'anello si instaura un meccanismo di contesa. La soluzione della contesa è basata sull'indirizzo delle stazioni. La stazione che ha inviato la PDU di tipo SOLICIT_SUCCESSOR ha il compito di risolvere la contesa e per far ciò utilizza PDU di controllo di tipo RESOLVE_CONTENTION. Alla PDU di tipo RESOLVE_CONTENTION sono associate quattro finestre temporali identificate dai quattro possibili valori che possono assumere i due bit più significativi dell'indirizzo delle stazioni. Le stazioni rispondono con un SET_SUCCESSOR in una delle quattro finestre temporali a seconda del loro indirizzo; la prima per le stazioni con i primi due bit a 1, la seconda per quelle 10, la terza per 01 e la quarta per 00. Le finestre temporali sono di lunghezza superiore al ritardo di propagazione, così che l'inizio di una trasmissione possa bloccare le trasmissioni successive (grazie ad un meccanismo di carrier sense). Ne consegue che le stazioni i cui due bit

più significativi di indirizzo sono uguali a 1 hanno priorità sulle altre. Se la contesa è risolta, ovvero solo una stazione è nel gruppo di quelle a massima priorità, si sceglie la stazione vincente e la si inserisce nell'anello, altrimenti si instaura una seconda fase di risoluzione mediante una nuova PDU RESOLVE_CONTENTION, in cui la risoluzione della contesa tra le stazioni con i primi due bit dell'indirizzo uguali a 1 è demandata ai secondi due bit di indirizzo, e così via.

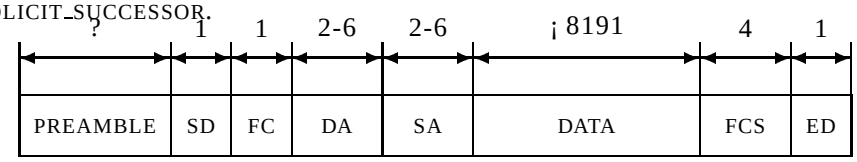
È evidente che le stazioni con indirizzo più alto entrano nell'anello logico prima delle altre.

L'algoritmo per l'inserzione di una stazione nell'anello logico è eseguito da una stazione che possiede il token e che, dopo aver completato la trasmissione delle proprie PDU dati, ma prima di passare il token, avendo ancora tempo a disposizione, prova a verificare la necessità di inserire nuove stazioni nell'anello logico. Ne consegue che le possibilità di inserimento di una stazione sono più elevate quando la rete è scarica.

Se la stazione che ha il possesso del token si guasta, o se manca l'alimentazione, o se si guastano due stazioni successive sull'anello logico, è necessario reinizializzare la rete.

Fase di inizializzazione della rete

La procedura di inizializzazione della rete è molto simile a quella utilizzata per l'inserimento di una nuova stazione e si basa sulla PDU di controllo di tipo CLAIM_TOKEN, inviata dalla prima stazione che si accorge della mancanza del token sulla rete. Se si ha colllisione nell'invio delle PDU CLAIM_TOKEN si usa l'algoritmo di risoluzione della contesa visto in precedenza. La stazione vincente crea poi l'anello logico inviando PDU di tipo SOLICIT_SUCCESSOR.



00000000	CLAIM_TOKEN
00000001	SOLICIT_SUCCESSOR 1
00000010	SOLICIT_SUCCESSOR 2
00000011	WHO_FOLLOWS
00000100	RESOLVE_CONTENTION
00001000	TOKEN
00001100	SET_SUCCESSOR

Figura 5.31. Formato di una PDU nello standard token bus e codici dei comandi di controllo del byte FC.

Il formato delle PDU è riportato nella figura 5.31. Il preambolo è utilizzato per la sincronizzazione di bit dei ricevitori. Il campo FC (Frame Control) è utilizzato per implementare tutti i meccanismi di controllo di accesso e i codici corrispondenti alle diverse PDU di controllo sono riportati nella figura.

5.9 Lo standard FDDI

Lo standard FDDI è nato dal tentativo di aumentare sia la velocità di trasmissione sia la dimensione massima della rete rispetto ai valori consentiti nelle LAN tradizionali. FDDI è stato il primo tentativo di realizzazione di reti ad alta velocità in grado di coprire un'area metropolitana. FDDI però non è riuscito ad imporsi come standard di rete metropolitana ed oggi è il principale standard per LAN ad alte prestazioni.

Lo standard FDDI è ottenuto con un miglioramento tecnologico dello standard IEEE 802.5. Esso prevede una topologia a doppio anello in fibra ottica, caratterizzata da un protocollo d'accesso di tipo control token, velocità di trasmissione di 100 Mbit/s, massima dimensione della rete pari a 200 km.

I due fattori fondamentali che hanno determinato la scelta dello standard 802.5 come punto di partenza per l'evoluzione verso velocità e distanze maggiori sono i seguenti.

- I protocolli CSMA hanno buone prestazioni quando il parametro a (ritardo normalizzato di propagazione sul canale) è piccolo; a è direttamente proporzionale sia alla dimensione della rete sia alla velocità di trasmissione sul canale, per cui l'uso dei protocolli CSMA non è ragionevole in reti ad alta velocità e di grandi dimensioni.

- La topologia ad anello si può facilmente implementare utilizzando tanti collegamenti punto-punto e quindi rigenerando il segnale in ogni stazione; in un bus invece il segnale tende a distorcersi mentre si propaga, e questo fenomeno è tanto più evidente quanto più la rete è di grandi dimensioni.

Le caratteristiche salienti dello standard FDDI sono:

- uso della fibra ottica come mezzo trasmissivo;

- topologia a doppio anello;

- codifica di linea $4B/5B$;

- velocità di trasmissione di 100 Mbit/s,

- protocollo d'accesso control token con modalità single token;

- uso di priorità per le PDU dati;

- possibilità di avere due tipi di stazioni diverse.

Per riuscire ad avere una implementazione hardware di complessità accettabile, si usano vari accorgimenti.

- Ogni stazione ha un suo clock interno; in questo modo non è necessaria una sincronizzazione globale della rete (contrariamente a quanto prevede lo standard 802.5). Per recuperare le differenze tra i clock, ogni stazione ha un buffer di 10 bit e il campo dati è limitato a 4500 byte.

- Al posto della codifica Manchester, che richiede una frequenza doppia rispetto alla velocità nominale di trasmissione, per tenere bassa la frequenza in linea si utilizza il codice $4B/5B$ che usa 5 intervalli di segnalazione per trasmettere 4 bit; di conseguenza il segnale di clock ha una frequenza di 125 MHz. Alcuni dei codici non utilizzati per trasmettere le sequenze di 4 bit sono utilizzati come caratteri di controllo (ad esempio esiste un carattere di IDLE).

Il doppio anello passa attraverso wiring concentrator, come nel caso dello standard 802.5. Se c'è un guasto su un canale punto-punto il doppio anello è riconfigurato come anello singolo, e la rete continua a funzionare. L'implementazione hardware permette di riconoscere dalla mancanza della portante un canale guasto permettendo alle stazioni di riconfigurarsi in modo opportuno. Esistono due tipi di stazioni: le stazioni di tipo A sono collegate ad entrambe gli anelli, mentre le stazioni di tipo B sono collegate ad un solo anello attraverso una stazione A che svolge la funzione di wiring concentrator. Le stazioni di tipo B in caso di guasto sono scollegate dalla rete.

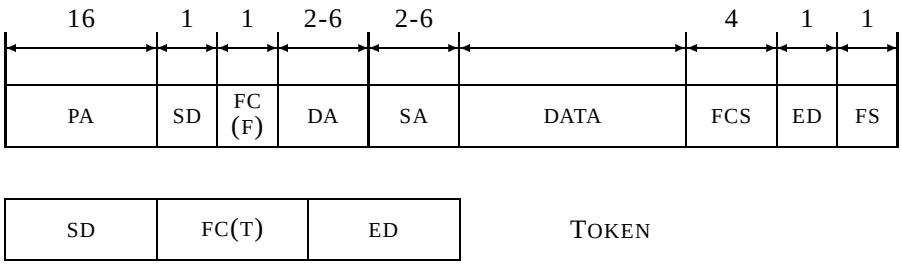


Figura 5.32. Formato di PDU FDDI

Il protocollo MAC utilizzato in FDDI è il protocollo control token con modalità single token, poiché la modalità single packet risulta troppo inefficiente a causa dell'elevata velocità di trasmissione e della grande lunghezza dell'anello.

I formati sia delle PDU sia del token sono molto simili a quelle adottate dal protocollo 802.5. I formati delle PDU e del token FDDI sono illustrati nella figura 5.32. Entrambi sono preceduti da 16 byte di preambolo PA e un byte di SD (Start Delimiter). Il campo FC (Frame Control), identifica la funzione della PDU (dati o token) e il campo ED (End Delimiter) termina la PDU. La PDU dati contiene anche i campi di indirizzo sorgente SA, indirizzo destinatario DA, un campo di informazione e un campo di stato FS.

Poiché la velocità di trasmissione prevista in FDDI è molto superiore rispetto al caso dello standard 802.5, può accadere che la stazione non riesca a trasmettere immediatamente i propri dati quando riceve il token; in questo caso la stazione può inserire una serie di simboli di IDLE fino a quando è pronta a trasmettere i primi bit della PDU.

Esistono due tipi di token; il primo è il token classico, utilizzato per distribuire i permessi di accesso alle diverse stazioni; quando una stazione dispone del token ed è quindi abilitata ad accedere al canale, può utilizzare un restricted token per interrogare le altre stazioni sull'anello.

Le priorità non sono legate a bit di informazione presenti all'interno del token, ma sono legate al carico esistente sulla rete, che è valutato in base al tempo di rotazione del token. Ogni stazione misura l'intervallo di tempo tra due istanti successivi in cui ha ricevuto il token. In base alla durata di questo intervallo potrà trasmettere PDU a un diverso livello di priorità. Se la rete è carica, il tempo di rotazione del token è più elevato, e si possono trasmettere solo PDU ad alta priorità. Se la rete è scarica le stazioni possono trasmettere PDU a qualunque priorità. In questo modo si attua una sorta di controllo del traffico inviato sulla rete; inoltre, le priorità sono stabilite in modo completamente distribuito, senza bisogno di un controllo centralizzato.

La mancanza di un controllo centralizzato in FDDI rende più complicata l'identificazione delle PDU spurie all'interno della rete, ovvero di quelle PDU che non sono state estratte dalla stazione che le ha inserite sull'anello.

5.10 Lo standard IEEE 802.6 o DQDB

Lo standard IEEE 802.6 riguarda le reti metropolitane (MAN), cioè reti pubbliche ad alta velocità e di dimensioni ragguardevoli. Lo standard è anche conosciuto con il nome DQDB (*Distributed Queue Dual Bus)*, che si riferisce alla topologia a doppio bus unidirezionale ed al protocollo di sottolivello MAC ad accodamento distribuito.

Lo standard DQDB trae le sue origini dalla proposta formulata da R. Newman con il nome QPSX.

Le caratteristiche principali dello standard DQDB sono:

- topologia a doppio bus unidirezionale,
- uso della fibra ottica come mezzo trasmissivo,
- velocità di trasmissione di 34 o 150 Mbit/s su ogni bus,
- protocollo d'accesso completamente distribuito,
- suddivisione dei tempi in slot di durata fissa,
- priorità sulle PDU dati.

Una rete DQDB è composta da due bus monodirezionali, sui quali si propagano slot di durata fissa, nei quali le stazioni possono inserire frammenti delle PDU da trasmettere (detti segmenti o celle); gli slot sono generati dalle due stazioni (dette *headend*) poste all'inizio dei due bus (la stazione 1 per il bus A e la 6 per il bus B nell'esempio riportato nella figura 5.33).

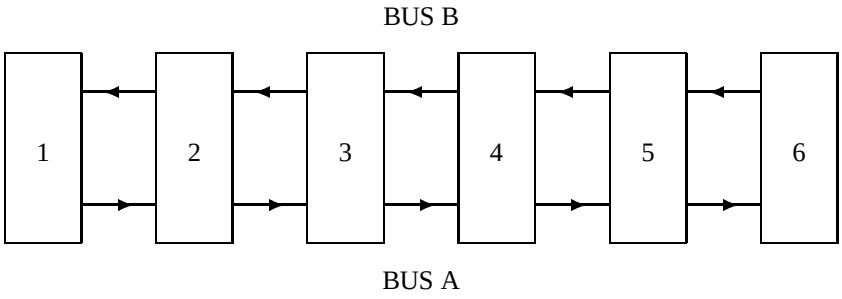


Figura 5.33. Topologia della rete DQDB

Le PDU di informazione da trasferire in rete, in generale di dimensioni variabili, devono essere inserite negli slot; per questo motivo è necessario disporre di un meccanismo di segmentazione (e di riassemblaggio) delle PDU in celle di dimensione tale da poter essere contenute in uno slot.

Ogni stazione è in grado di trasmettere e ricevere informazioni su entrambi i bus e la scelta del bus su cui trasmettere è basata sulla posizione relativa della stazione ricevente rispetto alla stazione trasmittente. In particolare, se si identificano le stazioni con un numero crescente da sinistra verso destra, ogni stazione utilizzerà il bus A per inviare PDU alle stazioni identificate da un numero superiore al proprio e il bus B per le stazioni con numero inferiore.

Il protocollo prevede un meccanismo di accodamento distribuito, basato su richieste (o prenotazioni) degli slot da parte delle stazioni. Gli slot contengono, oltre ad una parte utilizzabile per il trasferimento delle celle, una parte di controllo in cui è indicato se lo slot è libero o occupato ed in cui le stazioni inseriscono le richieste. Ogni stazione che vuole inserire una cella in uno slot deve iniziare una procedura di richiesta.

Il meccanismo di accesso è il seguente: supponendo che una stazione voglia trasmettere sul bus A,

- accoda una richiesta per la trasmissione in uno slot sul bus B;
- lascia passare sul bus A un numero di slot vuoti pari al numero di slot richiesti precedentemente dalle altre stazioni;
- accede al bus inserendo la cella nel primo slot libero che passa sul bus A.

Il processo solo a questo punto viene ripetuto per la cella successiva.

Il protocollo tenta di ricostruire in ogni stazione una copia della coda globale delle richieste di trasmissione di tutte le stazioni a valle, in modo tale da soddisfare le richieste di accesso al canale secondo una politica FCFS. Questo comportamento ideale si raggiungerebbe solo se il ritardo di propagazione fosse nullo. I ritardi di propagazione non nulli danno una immagine della coda globale ideale diversa alle varie stazioni, per cui in alcuni casi si possono verificare fenomeni di accaparramento delle risorse, come ampiamente documentato in letteratura.

5.10.1 Descrizione del protocollo di sottolivello MAC

I due head-end generano una sequenza di slot, ciascuno di dimensione pari a 53 byte, il cui formato è riportato nella figura 5.34.

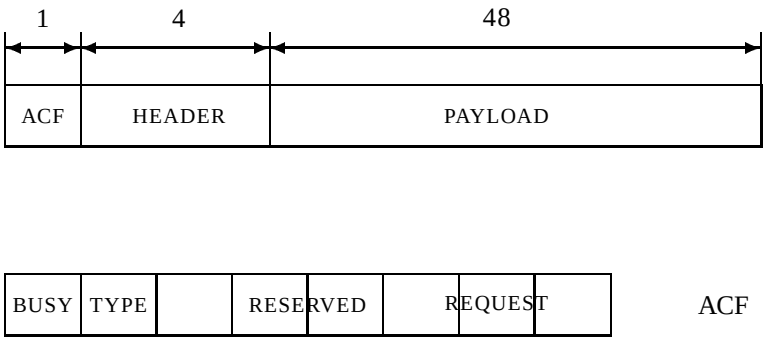


Figura 5.34. Formato di una slot DQDB.

La trasmissione sul canale è organizzata in trame della durata di 125 μs. Ogni trama contiene un numero intero di slot, il cui valore dipende dalla velocità della rete.

Esistono due tipi di slot:

- gli slot PA (Pre Arbitrated), che permettono la trasmissione isocrona, in particolare fino a 48 canali a 64kbit/s in una cella;
- gli slot QA (Queue Arbitrated) per la trasmissione asincrona, che contengono fino a 44 byte di dati.

Mentre gli slot PA sono preallocati e riservati a servizi isocroni, gli slot QA sono a disposizione delle trasmissioni a pacchetto e tutte le stazioni sono in competizione per l'uso di ogni slot. Il protocollo per accedere agli slot QA è il protocollo caratteristico dello standard DQDB.

I due bit fondamentali per il funzionamento del protocollo per gli slot QA sono il BUSY bit e il REQUEST bit del campo ACF. Il primo segnala se lo slot è libero o occupato da una cella, il secondo segnala se lo slot porta una richiesta. Uno slot quindi può trasportare contemporaneamente una cella e una richiesta. In realtà, come si vede dalla figura 5.34, il campo request si estende su tre bit: questo si spiega col fatto che vi sono tre livelli di priorità, quindi vi sono code diverse per ogni livello di priorità. Le stazioni sono dotate di un buffer di dimensioni sufficienti a contenere il solo campo ACF: quando arriva uno slot viene letto tale campo e si decide cosa fare. Il ritardo introdotto da ogni stazione è quindi modesto.

Senza perdere in generalità, esaminiamo il caso di trasmissione delle informazioni sul bus A e invio delle richieste sul bus B. Le estensioni al caso di traffico su entrambi i bus sono immediate. Trascuriamo per semplicità l'esistenza di tre livelli di priorità.

In ogni stazione esistono due contatori:

- un contatore RQ (*Request Counter*), che contiene il numero delle richieste non ancora soddisfatte inoltrate da tutte le altre stazioni prima che la stazione avesse un dato da trasmettere; chiaramente poiché stiamo esaminando il caso di trasmissione sul bus A, si tratta delle richieste inoltrate dalle stazioni situate "a valle", cioè a destra, rispetto alla stazione che stiamo considerando.

- un contatore CD (*CountDown counter*), che conta le richieste che la stazione deve soddisfare, ovvero gli slot vuoti che la stazione deve lasciare passare prima di potere inserire in uno slot la sua cella.

Inoltre esistono due code:

- una per le richieste,

- una per le celle.

Supponiamo che la stazione non debba trasmettere celle, cioè sia in uno stato di *idle.* Allora, al fine di mantenere aggiornata l'immagine della coda, l'algoritmo prevede che:

- ogni slot contenente una richiesta sul bus B provoca un incremento di una unità del contatore RQ;

- ogni slot vuoto sul bus A provoca una diminuzione di una unità del valore di RQ (se non è già zero): infatti uno slot libero che passa sul bus A verrà utilizzato da una delle stazioni seguenti per trasmettere una cella.

Quando una stazione ha una PDU da trasmettere:

- attua un algoritmo di segmentazione che introduce l'informazione di controllo necessaria e produce una serie di celle di lunghezza pari a 44 byte, e inserisce una richiesta nella coda delle richieste e una cella nella coda delle celle da trasmettere;

- nello stesso istante in cui inserice la richiesta e la cella nelle due code, copia il contenuto del contatore RQ nel contatore CD, e azzera il primo;

- decrementa il contatore CD ad ogni passaggio di uno slot vuoto sul bus A e quando CD raggiunge il valore 0 può utilizzare il primo slot libero per trasmettere la cella;

- incrementa il contatore RQ ad ogni richiesta che arriva sul bus B.

Se la stazione ha altre celle da trasmettere, ripete la sequenza appena descritta subito dopo aver trasmesso la cella, se no esegue l'algoritmo già visto per la condizione di idle.

Si osservi come le due code delle celle e delle richieste siano separate. In particolare, non è necessario attendere che la richiesta relativa alla cella in corso di trasmissione sia stata evasa (cioè sia trasmessa sul bus B) prima di inserire la cella in uno slot. Questa operazione è permessa anche se l'immagine della coda che ha la stazione in esame è in questo modo diversa dall'immagine che hanno le stazioni "a monte", in quanto ad esse la prenotazione della stazione considerata arriverà dopo quelle provenienti dalle altre stazioni. Questa difformità non influisce in modo grave sul funzionamento, dato che si tratta semplicemente di uno scambio di posizioni all'interno delle code. Alle stazioni infatti interessa solo sapere quanti slot vuoti devono lasciar passare prima di trasmettere, ma non sanno da chi verranno utilizzati tali slot. La stazione considerata inserirà i dati nello slot vuoto e le stazioni a valle, trovando tale slot occupato, non lo utilizzeranno. È invece necessario aspettare di avere trasmesso una cella prima di trasmettere quella successiva. Il vantaggio è la diminuzione dei tempi di accesso; in condizioni di carico basso le PDU sono inserite immediatamente sul canale (se non si hanno richieste pendenti) anche prima di fare la prenotazione.

L'uso delle priorità è semplice; per ogni priorità che si vuole gestire, in tutte le stazioni i contatori e le code descritte in precedenza sono replicati; sono presenti tre bit per le prenotazioni e quindi si possono gestire tre livelli di priorità.

5.10.2 Osservazioni sul funzionamento del protocollo

A causa dei ritardi di propagazione tra nodo e nodo e della elevata velocità di trasmissione, le richieste arrivano alle diverse stazioni in momenti diversi. L'immagine della coda globale è quindi diversa in ogni stazione e la gestione della coda non è esattamente FIFO. L'influenza sui ritardi medi delle PDU è piccola, ma la varianza si può modificare in modo significativo.

Un altro fenomeno spiacevole è quello che si verifica quando una stazione che deve trasmettere poco traffico è compresa tra due stazioni che devono trasmettere molto traffico; la stazione non riesce ad accedere alla risorsa trasmissiva, poiché non vede slot dati liberi sul bus su cui vuole trasmettere, ma neanche slot per le richieste liberi sull'altro bus perché i due bus sono accaparrati dalle due stazioni adiacenti. Questo problema causa una iniquità nel funzionamento del protocollo, che però si manifesta principalmente in condizioni di sovraccarico della rete. La versione definitiva dello standard prevede un meccanismo di *bandwidth balancing* per cui le stazioni lasciano passare uno slot dati libero ogni k anche se non esistono richieste pendenti. In questo modo ad ogni stazione non è concesso di occupare più di k slot liberi consecutivi, dove k dipende dalla configurazione della rete.

Il protocollo DQDB ha avuto un discreto successo per la sua semplicità e per le prestazioni significativamente migliori rispetto a FDDI. Il problema maggiore che si è riscontrato sta nella necessità di ricostruire le PDU a partire dalle celle; è necessaria, in ogni stazione, una gestione del riassemblaggio delle PDU separata per ogni utente della rete.

6

Interconnessione di reti locali

6.1 Dispositivi di interconnessione

Le reti locali hanno limiti in termini di dimensioni massime ammesse, carico massimo supportato e numero massimo di sistemi collegabili. Quando si vuole oltrepassare uno o più di questi limiti, bisogna creare una LAN estesa (a volte indicata con le sigle ELAN, XLAN o BLAN). Le possibili tipologie di interconnessione si distinguono in base al livello a cui si opera nella pila OSI. Se l'interconnessione riguarda il livello fisico si avranno i "**repeater**", se riguarda il livello data-link si avranno i **bridge**, i **router** per il livello network e i **gateway** per i livelli superiori al livello rete.

I **repeater** consentono il collegamento a livello fisico di due LAN omogenee, ossia caratterizzate dalla stessa interfaccia e dallo stesso protocollo d'accesso al mezzo trasmissivo. La topologia risultante non deve presentare anelli. In base al loro modo di operare si distinguono in:

- **bit repeater**: ricevono, rigenerano, risincronizzano e trasmettono il segnale. Introducono ritardi e si possono utilizzare in reti caratterizzate dalla stessa velocità di trasmissione.
- **buffer repeater**: introducono buffer e vengono utilizzati nell'interconnessione di LAN con velocità diverse. Non effettuano controllo di flusso.

I **bridge**, che sono apparati di livello 2, interconnettono LAN con livelli fisico e MAC differenti ma caratterizzate dallo stesso livello LLC. Essi trasmettono solo i pacchetti che devono effettivamente transitare da una LAN ad un'altra, mantenendo separati i traffici locali delle singole LAN che interconnettono. Questa loro funzionalità, detta di "*filtraggio*" (filtering), permette di ottenere un traffico globale sulla BLAN superiore a quello massimo ammesso per ogni singola LAN. Tale ritrasmissione avviene con una modalità di "*store & forward*", cioè il pacchetto è ricevuto dal bridge, che si limita solo a leggerlo e a ritrasmetterlo se è destinato a macchine residenti su LAN differenti rispetto a quella che l'ha generato. I bridge possono interconnettere LAN con lo stesso MAC oppure con MAC differenti. In questo secondo caso devono tradurre la PDU di livello 2, ricevuta da una LAN, nella PDU di livello 2 da trasmettere all'altra LAN.

In reti che utilizzano protocolli ad accesso casuale (ad esempio CSMA/CD), i bridge hanno sia la proprietà di rompere i domini di collisione, dove per dominio di collisione si intende la porzione di rete in cui due trame, trasmesse simultaneamente da due stazioni diverse, collidono, sia di trasmettere, con modalità "*store & forward*", solo i pacchetti che realmente devono transitare da una rete all'altra, mantenendo separati i traffici delle singole LAN. Con riferimento alla figura 6.1 a), le quattro stazioni connesse alla LAN, ad esempio una Ethernet a 10 Mb/s, appartengono allo stesso dominio di collisione: una trasmissione di A verso

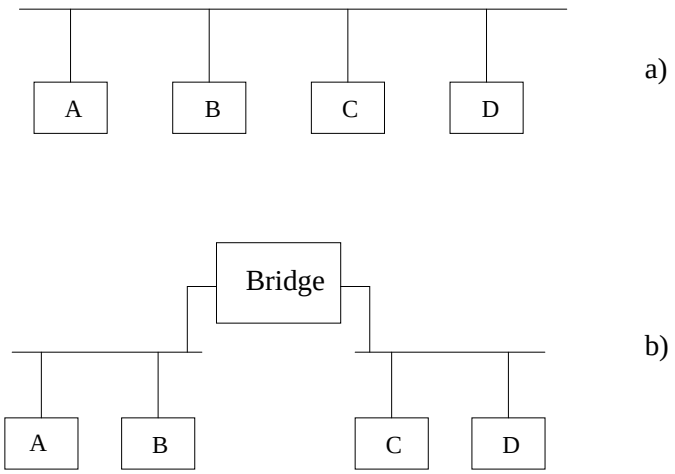


Figura 6.1. Utilizzo di un bridge per l'interconnessione di due LAN Ethernet

B occupa tutto il canale (10 Mb) e toglie, ad ogni altra stazione, la possibilità di trasmettere; anche C e D, se vogliono comunicare tra loro, sono costrette ad aspettare che il canale sia libero. Al contrario, se si suddivide lo stesso numero di stazioni su due reti Ethernet a 10 Mb, connesse tramite un bridge, [si veda la figura 6.1 b)], si realizzano due domini di collisione separati, permettendo così simultaneamente una comunicazione tra A e B e una tra C e D. Nel primo caso il traffico smaltito è di 10 Mb/s e la dimensione massima della rete, 2 km, nel secondo caso si ha un traffico smaltito di 20 Mb/s in condizioni di totale località di traffico ed una dimensione massima complessiva di circa 4 km (2 km per ogni spezzone); il bridge ha quindi permesso la creazione di una rete più estesa e con una capacità totale maggiore.

L'interconnessione di più LAN tramite bridge presenta le seguenti caratteristiche:

- si migliora l'affidabilità della rete (se si guasta una delle LAN le altre continuano a funzionare);
- si migliorano le prestazioni: separando il traffico locale da quello globale si sfrutta la diversità spaziale;
- si migliorano le caratteristiche di sicurezza: è possibile, se necessario, confinare il traffico contenente informazioni riservate;
- è possibile collegare reti distanti fisicamente utilizzando due half-bridge collegati tra loro con un canale punto-punto.

I **router**, cioè commutatori di pacchetto operanti a livello 3 del modello di riferimento OSI, garantiscono l'interconnessione tra reti eterogenee a livello MAC e LLC. Eseguono algoritmi di instradamento, ovvero scelgono il percorso ottimale tra la sorgente e la destinazione utilizzando opportuni algoritmi di instradamento. Analogamente ai bridge, memorizzano e ritrasmettono pacchetti (*store & forward*). L'instradamento nella rete Internet è basato sull'uso di tabelle di instradamento all'interno di router IP. Ciascun router sa verso

quale altro router deve indirizzare i pacchetti che riceve ma non ha conoscenza del percorso completo, né della posizione della sorgente e del destinatario. I router, a differenza dei bridge, permettono l'esistenza di anelli nella topologia.

Il *gateway* è utilizzato quando le architetture delle due reti che si vogliono interconnettere sono talmente diverse da richiedere di risalire a livelli superiori al livello rete, spesso addirittura fino al livello applicativo.

6.2 Interconnessione mediante bridge

I bridge hanno le seguenti caratteristiche generali:

- operano al livello 2 del modello di riferimento OSI, e più precisamente al sottolivello MAC; per questo sono molto spesso detti MAC-Bridge,
- hanno algoritmi di instradamento molto semplici: ogni bridge calcola autonomamente le sue tabelle di instadamento senza interagire con gli altri bridge, con un algoritmo di routing isolato;
- si utilizzano normalmente per le interconnessioni locali, anche se sono stati usati nel passato, in modo un poco problematico, anche per le interconnessioni geografiche.

I bridge possono essere realizzati secondo due filosofie diverse che differiscono per il luogo dove sono memorizzate le tabelle di instradamento (o tabelle di filtraggio):

- **transparent bridge**: sono i bridge conformi allo standard IEEE 802.1d, di derivazione Ethernet. Hanno le tabelle di instradamento a bordo e sono trasparenti, nel senso che i sistemi interconnessi alle LAN ignorano la loro esistenza;
- **source routing bridge**: sono i bridge di derivazione token-ring. Non hanno tabelle di instradamento a bordo, che sono invece mantenute dai sistemi connessi alle LAN; in fase di trasmissione del pacchetto, le stazioni devono specificare esplicitamente il cammino che il pacchetto dovrà fare per giungere a destinazione, indicando tutti i bridge da attraversare (che quindi sono indirizzati esplicitamente).

Vengono qui di seguito descritti brevemente i due diversi modi di realizzazione.

6.3 Transparent Bridge

Sono conformi allo standard IEEE 802.1d e servono per l'interconnessione di LAN IEEE 802 (si veda la figura 6.2) comprese le LAN Ethernet.

6.3.1 Struttura fisica e tabelle di instradamento

I bridge sono formati da una CPU general purpose, da due o più interfacce per l'interconnessione con le LAN e da un filtering database (una tabella di instradamento) contenuto in una memoria RAM. Ogni riga della tabella è formata da una coppia "*indirizzo destinazione/interfaccia di uscita*", che ne costituisce una riga (o entry). Le entry sono di due tipi:

- **statiche:** cioè configurate, tramite operazioni di management dal gestore;
- **dinamiche:** cioè inserite autonomamente dal bridge (*learning process*).

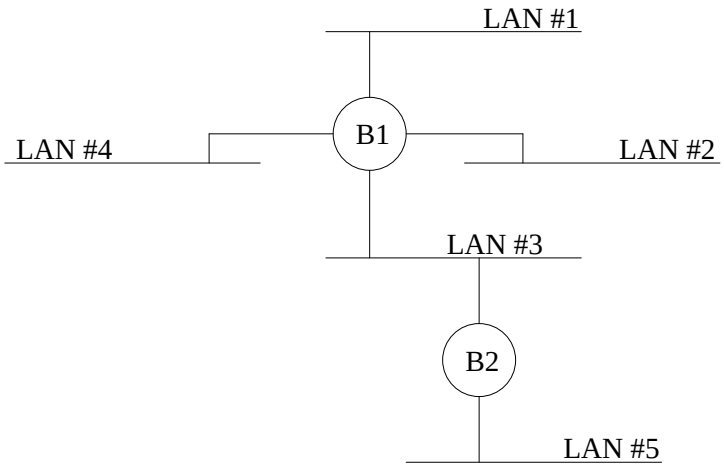


Figura 6.2. Generica interconnessione di LAN Ethernet tramite trasparent bridge

Il bridge, infatti, memorizza, per ogni trama ricevuta, la porta di arrivo e il mittente, informazioni che gli serviranno quando dovrà instradare pacchetti che hanno quella stazione come destinazione. Le entry statiche hanno priorità su quelle dinamiche: non verranno quindi mai create entry dinamiche per un indirizzo MAC di cui vi sia già un'entry statica. Le entry dinamiche hanno un tempo di vita limitato: ad ognuna è associato un timeout entro il quale la entry deve essere aggiornata o eliminata, questo per impedire che eventuali cambiamenti della rete rendano le tabelle inconsistenti. È preferibile che il bridge non conosca come raggiungere una destinazione piuttosto che instradi le trame secondo informazioni vecchie e magari non più valide a causa di guasti o riconfigurazioni.

Quando una trama arriva su un'interfaccia, la CPU ne analizza l'intestazione per individuare l'indirizzo MAC di destinazione; è importante osservare che i pacchetti non sono mai indirizzati direttamente ai Trasparent Bridge, di cui è ignorata l'esistenza (da qui il nome di bridge "trasparenti"), ma sempre alle stazioni di utente. L'indirizzo destinazione è confrontato con tutte le entry della tabella di instradamento; se è rilevata una corrispondenza, la CPU inoltra la trama unicamente sulla porta associata, in caso contrario la trama è inoltrata su tutte le porte attive in forwarding, tranne quella di provenienza.

Come detto i bridge devono instradare pacchetti sulla rete e quindi hanno bisogno di costruirsi tabelle di instradamento. Se la topologia della BLAN è ad albero, la costruzione di tali tabelle può avvenire con un algoritmo molto semplice, in modo automatico, tramite un processo di apprendimento (learning process).

Poiché è tuttavia preferibile avere topologie magliate per ragioni di affidabilità, occorre integrare il learning process con un algoritmo detto di spanning tree per riportare dinamicamente una topologia magliata ad una topologia ad albero, escludendo dall'operatività opportune porte di opportuni bridge. Tale problema non esiste nei source routing bridge, in quanto il pacchetto, quando viene generato, contiene la specifica completa del cammino che dovrà seguire.

Le funzioni fondamentali svolte da un bridge trasparente sono quindi:

- ricezione, filtraggio, processo di apprendimento e inoltro dei pacchetti;
- mantenimento delle informazioni necessarie per prendere le decisioni di filtraggio;
- governo e controllo della correttezza delle precedenti funzioni (management).

Vediamo brevemente le funzioni del primo punto.

6.3.2 Ricezione dei pacchetti (Frame Reception)

Esiste un database che associa gli indirizzi MAC delle stazioni di utente collegate alla rete, alle "porte" (connessioni) del bridge ad una LAN. L'entità MAC associata ad ogni porta riceve ed esamina tutti i pacchetti trasmessi sulla LAN cui è connessa.

La prima analisi riguarda il campo FCS per determinare se il pacchetto è corretto o errato. I pacchetti errati sono scartati.

I pacchetti indirizzati effettivamente alle entità di livello superiore del bridge (che sono normalmente una piccola parte) vengono affidati al livello LLC associato alla porta di ricezione. Questi pacchetti contengono come inidirizzo MAC o l'indirizzo di una porta del bridge o un indirizzo multicast cui appartiene almeno una porta del bridge.

Gli altri pacchetti sono passati all'entità MAC di inoltro.

6.3.3 Filtraggio dei pacchetti (Filtering Database)

I pacchetti trasmessi da un sistema S1 verso un sistema S2 vengono confinati dai bridge nelle LAN che formano il percorso da S1 a S2. Questo tipo di filtraggio è il più comune e serve a ridurre il traffico globale in rete.

Le funzioni principali che riguardano il mantenimento delle informazioni di filtraggio sono essenzialmente:

- l'apprendimento automatico (*learning process*) delle informazioni relative al filtraggio dinamico, attraverso l'osservazione del traffico della BLAN;
- definizione dell'età massima (*ageing time*) delle informazioni relative al filtraggio che sono state apprese automaticamente, oltre la quale le informazioni stesse vengono invalidate;
- calcolo e configurazione della topologia logica della BLAN (tramite l'algoritmo detto "*spanning tree*" che sarà presentato successivamente).

6.3.4 Inoltro dei pacchetti (Frame Forwarding)

Un pacchetto ricevuto su una porta di un bridge viene affidato al processo di inoltro che deve deciderne un eventuale accodamento per la trasmissione su altre porte. Condizione necessaria è che sia la porta di ricezione sia le porte di destinazione si trovino in stato di forwarding.

Il processo di inoltro accoda il pacchetto su una singola porta se questo ha un indirizzo di destinazione MAC di tipo singolo, su tutte le porte se tale indirizzo è multicast o broadcast.

Il processo di inoltro consulta la tabella di instradamento per determinare su quale porta eventualmente accodare il pacchetto in funzione del suo indirizzo di destinazione. Tale accodamento rispetta rigorosamente l'ordine di arrivo dei pacchetti, opera cioè in modalità FIFO (First-In First-Out). Un pacchetto viene rimosso da tale coda quando viene trasmesso, indipendentemente dall'esito dell'operazione, nel caso in cui venga superato il tempo massimo di transito del pacchetto nel bridge (*maximum bridge transit delay*) e quando la porta in considerazione abbandona lo stato di forwarding.

6.3.5 Processo di apprendimento (Learning Process)

Il processo di apprendimento osserva l'indirizzo sorgente dei pacchetti ricevuti su ogni porta e aggiorna le entry dinamiche della tabella di instradamento, condizionatamente allo stato delle porte.

Il MAC-SAP indica al processo di apprendimento che la stazione con quell'indirizzo è raggiungibile attraverso la porta che ha ricevuto il pacchetto. Tale metodologia di apprendimento è conosciuta come "routing isolato-backward learning" in quanto un indirizzo di sorgente attuale crea o aggiorna una entry dinamica della tabella di instradamento relativamente ad una destinazione che potrà essere utilizzata in fututo. Le condizioni in cui è possibile creare o aggiornare una entry dinamica sono:

- la porta da cui è stato ricevuto il pacchetto deve essere in uno stato che permetta l'apprendimento dell'indirizzo MAC (stato di learning o di forwarding);
- non esiste già un'entry statica (una entry fissata staticamente che non puó cambiare durante il processo) per quell'indirizzo MAC.

Se il numero risultante di tutte le entry supera la capacità massima della tabella di instradamento, una entry più vecchia viene rimossa per fare spazio alla nuova entry. Si ricordi che esiste anche il meccanismo del timeout per tenere le tabelle compatte.

6.3.6 Algoritmo di spanning tree

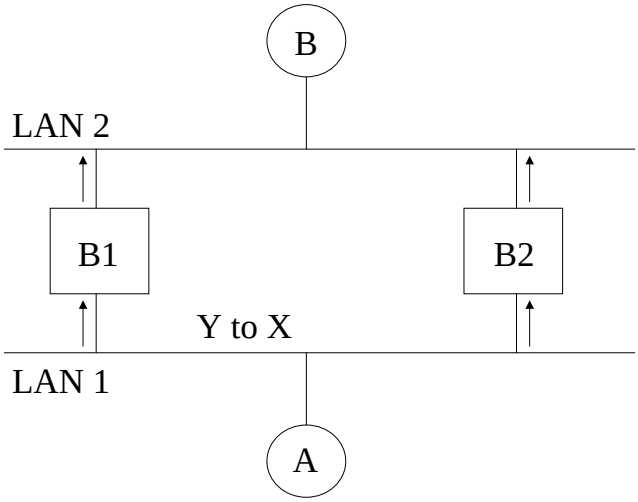


Figura 6.3. Presenza di un loop

L'algoritmo di spanning tree riconfigura una topologia magliata di una BLAN in una topologia ad albero, eliminando gli anelli, mediante la disabilitazione di alcune porte dei bridge, nel caso in cui ci siano più percorsi alternativi (si avrebbe altrimenti un fenomeno di duplicazione dei pacchetti). Si osservi infatti cosa accadrebbe se ci fosse un anello (si veda la i figura 6.3). Si supponga che A, appartenente alla LAN1,

trasmetta un pacchetto a B, appartenente alla LAN2. Entrambi i bridge, B1 e B2, ritrasmettono il pacchetto sulla LAN2: in tal modo B riceverà due copie del pacchetto, una da B1 e l'altra da B2. Supponiamo che B1 trasmetta il pacchetto che ha come inidirizzo sorgente A e come indirizzo destinazione B prima di B2; il pacchetto sarà ricevuto sulla LAN1 tramite B2 che, prima di B2, che che penserà che A si trovi nella LAN1. La destinazione B riceverà un'altra copia generata da B1; da questo momento in poi A non riceve più pacchetti poiché entrambi i bridge credono di essere collegati alla LAN2.

In caso di guasto sul percorso primario, lo spanning tree deve inoltre riconfigurare automaticamente la topologia della BLAN, senza la formazione di anelli in transitorio. Per gestire la configurazione della topologia attiva, l'algoritmo di spanning tree prevede l'assegnazione di una priorità ai bridge e alle porte di ciascun bridge. Tutti i bridge devono inoltre avere un identificatore univoco. A tal fine si definisce un "*bridge ID*" (identifica la priorità del bridge), un "*port ID*" (identifica la priorità della porta di un bridge). È essenziale anche la definizione di un indirizzo multicast che identifichi tutti i bridge del sistema. I valori più bassi di tali identificatori indicano priorità maggiore. La configurazione di una topologia attiva (albero) partendo da una topologia arbitraria (maglia) avviene ponendo alcune porte di alcuni bridge in blocking state (stato di disabilitazione temporanea). Le porte che sono in blocking state non partecipano alla topologia attiva, ma sono pronte ad entrare a farne parte in caso di guasto di qualche componente della BLAN.

Passi dell'algoritmo di spanning tree

L'algoritmo opera nei seguenti passi:

- **elezione del bridge radice (root bridge)**: poiché si vuole identificare un albero, il primo passo consiste nell'identificare la radice dell'albero. Per definizione è il bridge con identificativo (bridge ID) minore;
- s**elezione della porta del bridge radice (root port)**: per ogni bridge si identifica la porta più conveniente per interconnettere il bridge verso il bridge radice;
- **selezione del bridge designato (designated bridge)**: per ogni LAN si sceglie quale bridge è designato a interconnettere la LAN con il root bridge. Questo passo è particolarmente importante quando esistono più cammini tra la LAN e il bridge radice. Ogni LAN ha solo un bridge designato che è il bridge più vicino al bridge radice (ha costo minore) e che si incaricherà di trasmettere i pacchetti verso il bridge radice. A parità di costo si sceglie il bridge con bridge ID minore. La porta del bridge designato che interconnette la LAN è detta porta designata (designeted port). Il bridge radice è l'unico bridge che ha tutte porte designate.

Vediamo brevemente il modo di funzionamento.

La comunicazione tra i bridge per l'esecuzione dell'algoritmo avviene mediante lo scambio continuo di BPDU contenenti:

- identificatore del bridge sorgente (chi ha trasmesso la BPDU-*Bridge Protocol Data Unit*);
- identificatore del bridge considerato radice;
- identificatore della porta considerata come root port dalla sorgente (e relativo costo).

Inizialmente tutti i bridge trasmettono BPDU dichiarandosi bridge radice. Su ogni LAN dopo breve tempo solo il bridge con identificativo inferiore crede di essere radice; questo anche sulla LAN su cui è collegato il vero bridge radice. Quando un bridge collegato alla LAN su cui è presente il bridge radice riceve la BPDU con indicazione del bridge radice, è in grado di determinare la sua porta verso la radice ed il costo per raggiungere la radice (pari al costo della porta cui è collegato al bridge radice). Questi bridge

trasmettono BPDU contenenti informazioni sul bridge radice e sul loro costo per raggiungerlo. Basandosi su queste informazioni gli altri bridge calcolano il loro costo verso il bridge radice (sommando al costo che ricevono il costo della loro porta) e la loro porta radice. Al termine di questo processo tutti i bridge conoscono il bridge radice, e la loro distanza dal bridge radice. Su ciascuna LAN, diventa bridge designato il bridge con costo minimo verso il bridge radice. A parità di costo viene considerato più "vicino" alla radice il bridge con identificatore più basso; sullo stesso bridge, a parità di costo di porte, ha la precedenza la porta con identificatore più basso. Determinato ciò si procede alla messa in stato di blocking delle porte che non sono né *root port* né *designated port*. Fatto ciò si è determinata la topologia attiva e si può procedere alla trasmissione dei pacchetti.

6.4 Source routing

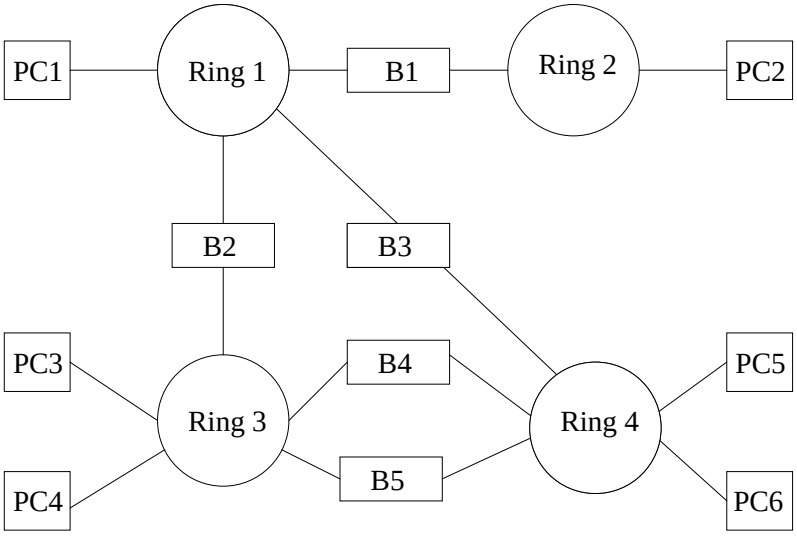


Figura 6.4. Interconnessione di LAN IEEE 802.5 tramite bridge source routing

I bridge source routing sono stati sviluppati per operare tra LAN IEEE 802.5 in uno scenario simile a quello della figura 6.4.

L'instradamento del messaggio è calcolato a priori dal sistema mittente ed è incluso in ogni pacchetto, il bridge non ha quindi bisogno di avere a bordo le routing table, che al contrario sono mantenute dalle stazioni di utente.

Ogni sorgente specifica il cammino completo che le trame devono seguire: inserisce un campo di informazione di instradamento (*RI - Routing Information*) dopo gli indirizzi di sorgente e destinazione. Osservando la figura 6.4, se PC1 vuole comunicare con PC6 deve specificare in tale campo che intende far passare il pacchetto attraverso i bridge B2 e B5 oppure attraverso B2 e B4 oppure attraverso B3. I sistemi mittenti devono quindi mantenere una tabella di instradamento contenente le destinazioni con cui sono interessati a

comunicare e che richiedono l'attraversamento di bridge. Le entry delle tabelle di instradamento vengono calcolate automaticamente tramite un processo di route discovery. Questo processo permette al sistema mittente di scoprire dinamicamente il percorso per raggiungere il sistema destinatario utilizzando pacchetti di esplorazione della BLAN (*ARE packets - All Routes Explorer packets*). Ogni bridge che inoltra un pacchetto multicast di route discovery ARE aggiunge al campo RI il suo identificativo (in modalità flooding). In questo modo l'informazione di instradamento viene costruita dai bridge al momento in cui il pacchetto ARE viene inoltrato da un segmento di LAN ad un altro. I pacchetti in loop nella rete possono essere eliminati ispezionando la strada già percorsa dai pacchetti. Quando il pacchetto ARE raggiunge la stazione destinazione, esso contiene nel suo campo RI tutte le informazioni di percorso (LAN-Bridge) che indicano quindi su quale LAN si deve ritrasmettere la trama e quale bridge si deve occupare del successivo instradamento. La stazione destinazione lo ritrasmetterà alla stazione mittente che sceglierà il cammino a lei più conveniente tra tutti quelli ricevuti dalla destinazione e lo inserirà nella relativa entry della sua tabella.

Questa tecnica ha il vantaggio di non richiedere tabelle di instradamento nei bridge, in quanto queste sono mantenute ed utilizzate dai sistemi mittenti, ma:

- complica il software delle stazioni utente che devono occuparsi di determinare le possibili strade per raggiungere il destinatario;
- obbliga ad indicare l'intero percorso in ogni trama diminuendo l'efficienza di trasmissione delle informazioni.
- flooding per conoscere la strada.

Per le reti che adottano il source routing sono possibili quattro tipi di instradamento:

- **null**: si trasmette il pacchetto sulla rete locale senza richiedere instradamento. Viene utilizzato quando si deve trasmettere un pacchetto destinato a un sistema connesso alla stessa LAN su cui è stato generato e quindi verrà ignorato dai bridge;
- **non-broadcast**: il pacchetto segue l'unica strada possibile, indicata nelle informazioni di routing nell'intestazione del pacchetto. Un pacchetto con routing non-broadcast è letto e trasmesso solo se il bridge compare nella lista di instradamento contenuta nell'intestazione del pacchetto. Arriverà una sola copia alla destinazione.
- **all-routes broadcast**: si trasmette su tutte le possibili strade. Ogni bridge instrada su tutte le LAN a cui è collegato. Si creano copie multiple, una per ogni possibile strada;
- **single-route broadcast**: il frame compare una ed una sola volta in ogni LAN (si usa l'algoritmo spanning tree); il destinatario ne riceve una copia.

Gli instradamenti broadcast servono alle stazioni utente per individuare i possibili percorsi verso il destinatario. Si possono utilizzare due tecniche:

- la sorgente trasmette all-routes broadcast. Il destinatario manda un messaggio non broadcast con indicazione del percorso seguito da ogni frame che riceve. La sorgente sceglie quindi il percorso;
- la sorgente trasmette single-route broadcast. Il destinatario risponde usando all-routes broadcast, ottenendo in tal modo tutte le possibili strade (meccanismo di route discovery ARE spiegato precedentemente). Una volta ottenute tutte le possibili strade si può scegliere la strada che si vuole utilizzare secondo diversi criteri:

  - minimo numero di reti (o bridge) attraversati;
  - costo minimo, dove il costo è inversamente proporzionale, ad es., alla velocità della rete;
  - la strada per la quale il messaggio di risposta arriva prima (stimando il ritardo);
  - in modo casuale.

Un pacchetto con instradamento all-routes è ricevuto dal bridge che:

- aggiunge il proprio identificativo nella lista di instradamento contenuta nell'intestazione;
- trasmette il pacchetto su tutte le LAN su cui è collegato, evitando di ripetere la trasmissione se il pacchetto è già stato trasmesso dal bridge stesso.

Un pacchetto con instradamento single-route può essere inoltrato solo seguendo il funzionamento previsto nell'algoritmo spanning-tree.

I tre tipi di instradamento sono "ortogonali" rispetto al tre tipi di indirizzo presenti nelle reti locali a livello MAC: individual, group e broadcast.

6.4.1 Confronto tra spanning tree e source routing

L'algoritmo **Spanning tree** presenta le seguenti caratteristiche:

- concentra il traffico sulla radice dell'albero;
- obbliga ad un sottoutilizzo delle risorse: si sfrutta poco la diversità spaziale perché tipicamente i pacchetti fanno cammini più lunghi di quelli ottimi (anche se spesso le LAN hanno struttura gerarchica che riflette anche la distribuzione del traffico);
- i bridge devono gestire le tabelle di instradamento;
- il concetto di costo permette configurazioni predefinite della topologia.

L'algoritmo **Source routing** presenta i seguenti svantaggi:

- non è completamente trasparente (le stazioni devono essere modificate);
- aumenta la dimensione delle trame;
- si adatta meno alle variazioni topologiche;
- il processo di ricerca dell'instradamento può congestionare la rete, ad esempio nel caso di più bridge paralleli tra le stesse LAN;
- difficile sfruttare i vantaggi potenziali in diversità spaziale: le stazioni dovrebbero coordinarsi tra loro per sfruttare tale possibilità;
- richiede modifiche ed estensioni al protocollo token ring (ricalcolo del FCS, i bridge devono modificare i bit A e C anche se l'indirizzo di destinazione non è il loro, si deve fornire alta priorità ai bridge?)

L'algoritmo più usato è sicuramente lo Spanning tree.

7

Reti locali ad alta velocità

7.1 Introduzione

La continua ricerca di soluzioni tecnologiche che consentano di soddisfare, da un lato le esigenze di una maggior banda e aumento dell'affidabilità e dall'altro una progressiva diminuzione dei costi, ha portato numerosi sviluppi nell'ambito delle tecnologie per reti locali.

In questo capitolo si discuterà principalmente delle evoluzioni delle reti locali tradizionali, e in particolare delle evoluzioni della rete Ethernet.

7.2 Evoluzione delle LAN cablate

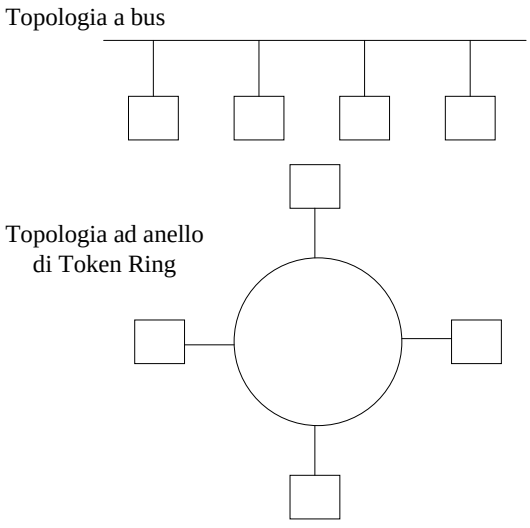


Figura 7.1. Topologia a bus e topologia ad anello

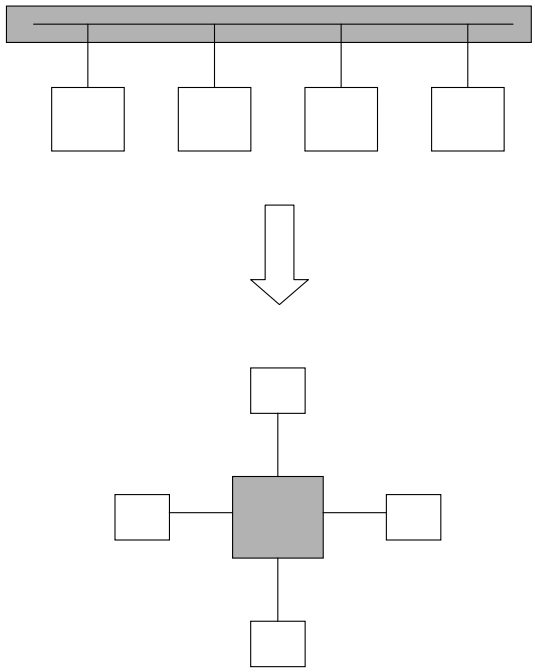


Figura 7.2. Trasformazione di topologie mediante un hub

Una rete locale prevede la condivisione da parte di più stazioni di un unico canale trasmissivo caratterizzato da alta velocità e basso tasso di errore. Questa descrizione si adatta molto bene alla struttura tipica di una rete Ethernet, che sfrutta una topologia a bus, e alla topologia ad anello di Token Ring (si veda la figura 7.1). Da cablaggi di tipo bus, o anello, è possibile ricondursi a topologie stellari, il cui centrostella è costituito essenzialmente da un hub (si vedano la figura 7.2 e la figura 7.3). Questa situazione è detta *Collapsed Backbone*: la dorsale è collassata su unica scatola, l'*hub* appunto. Il centrostella può essere eventualmente ridondato per aumentare la tolleranza ai guasti. I vantaggi che si ottengono con una topologia stellare sono la semplicità sia di gestione sia di cablaggio. In realtà, se il centro stella è semplicemente un hub, la capacità trasmissiva del concentratore continua ad essere pari a quella del singolo collegamento, visto che si continua ad avere piena condivisione della banda. È però possibile sostituire gli hub con apparati di livello 2, detti *switch*, che permettono la trasmissione contemporanea di pacchetti che hanno coppie mittente-destinatario distinte. Ad esempio, se si vogliono interconnettere tramite switch 30 stazioni di una rete Ethernet standard a 10 Mb/s, posso arrivare ad avere 10 Mb/s effettivi per ognuna delle 15 coppie possibili, se in ogni istante ho coppie disgiunte di interlocutori; in questo caso la banda non è più condivisa, ma dedicata. In base a queste considerazioni e alle innovazioni tecnologiche degli ultimi anni, le reti Ethernet di nuova generazione, che verranno analizzate in dettaglio nei paragrafi seguenti, sono:

- **Switched Ethernet**
- **Fast Ethernet**
- **Gigabit Ethernet**

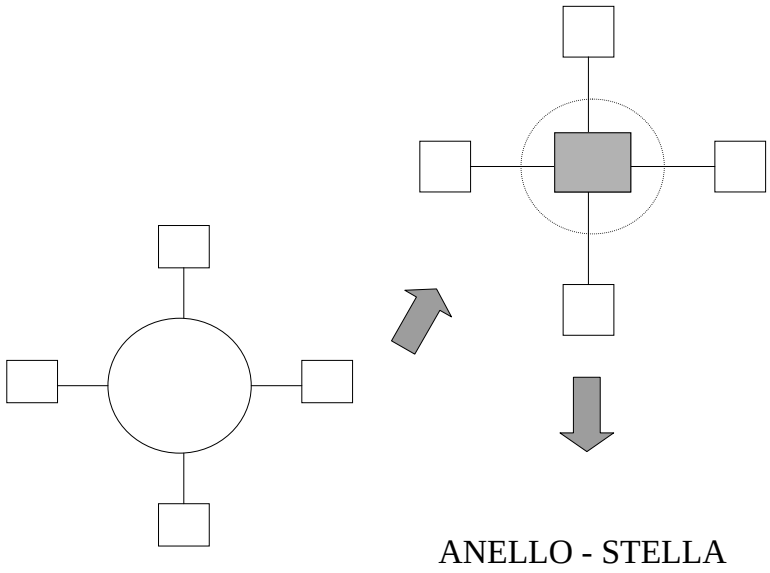


Figura 7.3. Trasformazione di topologie mediante un hub

7.3 Switched Ethernet

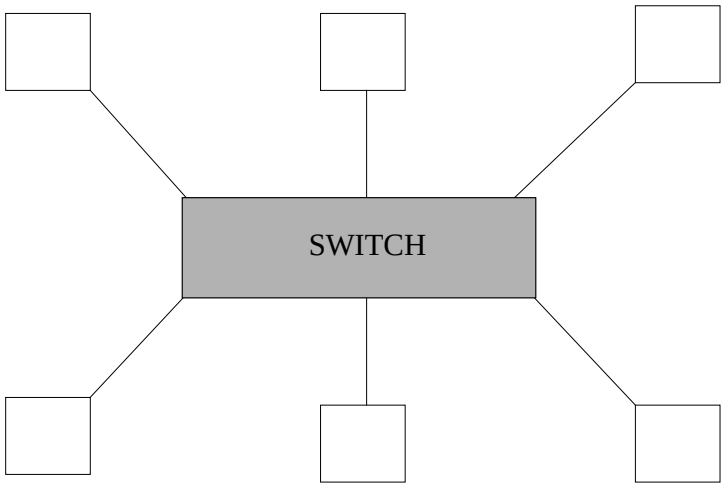


Figura 7.4. Switched Ethernet

Per Switched Ethernet si intende una rete Ethernet con Collapsed Backbone, il cui centrostella è costituito da switch. Lo switch Ethernet è analogo a un bridge, ma con un numero di porte nettamente superiore e ogni porta è dedicata ad un'unica stazione (si veda la figura 7.4). Questa topologia stellare fa sì che molti costruttori realizzino switch che non supportano l'algoritmo di Spanning Tree, algoritmo al contrario indispensabile con topologie magliate. Un'altra variante che viene introdotta negli switch riguarda la modalità di inoltro dei pacchetti. Esistono sostanzialmente tre modalità di Ethernet switching:

- **Store & Forward**: è la tecnica base con cui funzionano i bridge: il pacchetto è ricevuto completamente e poi ritrasmesso verso la destinazione. Si effettua il controllo sull'FCS (controllo d'errore) per l'eliminazione dei pacchetti corrotti.
- **Cut Throught (**o **On The Fly Switching)**: mentre riceve una trama MAC, lo switch esamina immediatamente l'indirizzo di destinazione, consulta la sua tabella di instradamento e se la porta di destinazione è libera, inizia a ritrasmettere la trama senza attenderne l'immagazzinamento completo; la decisone di inoltro è quindi presa durante il transito della trama. In questa modalità il controllo dell'FCS non è effettuato.
- **Fragment Free:** prima di iniziare a ritrasmettere il pacchetto si aspetta comunque un tempo pari alla *collision window*, garantendo così di non ritrasmettere frammenti di pacchetti collisi.

La modalità naturale per uno switch è la modalità Cut Throught; al contrario di un bridge, quindi, non si evita la propagazione di trame corrotte nella rete. Lo switch, però, esegue a posteriori un controllo sulla correttezza dei dati inoltrati al fine di disabilitare la modalità Cut Throught sulle porte con elevato tasso di errore. Il Cut Throught Switching ha il vantaggio di ridurre i tempi di latenza, ma può essere utilizzato solo se valgono le seguenti condizioni:

- su tutte le porte è presente lo stesso tipo di MAC;
- tutte le porte hanno la stessa velocità trasmissiva;
- la porta di destinazione è libera;
- la trama non è broadcast o multicast.

In tutte le altre situazioni è necessario fare *Store & Forward.* È ancora interessante osservare che in caso di pacchetti corti tutte e tre le modalità di Switching sono praticamente equivalenti.

7.4 Ethernet half e full duplex

Una rete locale è una struttura intrinsecamente half duplex nel senso che non è consentita la trasmissione contemporanea nelle due direzioni, ma si può trasmettere una sola stazione per volta. In realtà l'utilizzo di topologie stellari, con centrostella realizzato tramite switch, ha ridotto molto le problematiche relative alla condivisione del mezzo trasmissivo: avendo solo collegamenti *punto-punto*, le collisioni possono verificarsi unicamente tra la stazione e lo switch; lo switch spezza i domini di collisione. In realtà è possibile fare qualcosa di più: per la comunicazione punto-punto tra stazione e switch si possono utilizzare in modo unidirezionale due canali Ethernet tradizionali half duplex in parallelo, ottenendo complessivamente un canale Ethernet dedicato full duplex. Tecnologie full duplex sono quindi applicabili solo se il centrostella è costituito da switch, non da hub (si veda la figura 7.5). Mezzi trasmissivi full duplex consentono alle stazioni di trasmettere contemporaneamente; dato che le trasmissioni avvengono su canali fisici diversi, non sono soggette

Figura 7.5. Tecnologie full duplex

a collisioni. I limiti di distanza nella realizzazione di reti locali in tecnologia full duplex non sono quindi più dati dal livello MAC, ma esclusivamente da problemi legati al livello fisico, come, ad esempio, l'attenuazione dei cavi e il rumore introdotto. Tecnologie full duplex sono utilizzate in particolare per interconnettere punto-punto due switch o due bridge, per cui sono molto usate nelle dorsali.

7.5 Considerazioni

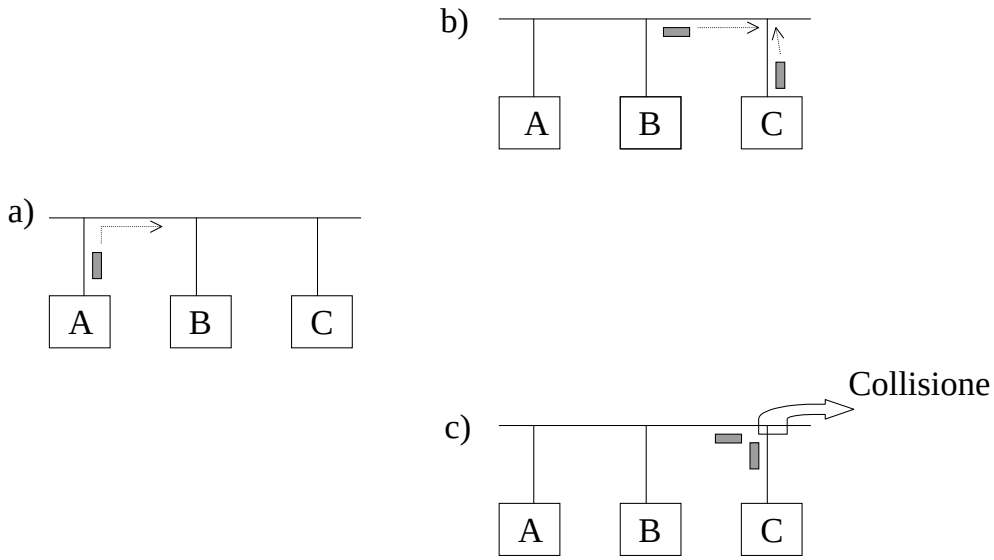


Figura 7.6. Meccanismo della collisione

La velocità in una rete non è un parametro indipendente, ma è legata alla lunghezza minima del pacchetto e alla distanza tra le due stazioni estreme della rete. L'algoritmo CSMA/CD prevede che per rilevare una collisione la stazione che sta trasmettendo debba continuare ad ascoltare il canale per tutto il tempo della trasmissione. Se durante tale trasmissione non si rileva alcuna collisione, la stazione assume che il pacchetto sia andato a buon fine. È necessario quindi progettare la rete in modo tale che non ci sia più rischio di collisione dopo che la stazione ha terminato di trasmettere. Questo progetto è fatto in base al pacchetto di lunghezza minima, che per Ethernet a 10Mbit è di 64 byte. Con riferimento alla figura 7.6, supponiamo che la stazione A trasmetta un pacchetto di lunghezza minima e che questo vada in collisione con C. A e C devono entrambe accorgersi della collisione, il caso più sfortunato è che C inizi a trasmettere poco prima che il pacchetto generato da A si sia propagato fino a lei. Allora nel punto in cui si trova C ho una collisione: mentre C la rileva immediatamente, A deve attenderne la propagazione all'indietro. Rilevare una collisione nel caso peggiore, quindi, è equivalente ad attendere che un pacchetto trasmesso da A, arrivi fino a C e ritorni ad A. In realtà, per poter trasmettere la sequenza di jamming, A deve accorgersi della collisione prima che la sua trasmissione dati abbia avuto termine. Il tempo impiegato da un pacchetto per andare e tornare da un estremo all'altro della rete è detto Round Trip Time e in base ad esso è possibile definire una finestra temporale, detta *collision window*, la quale assicura di rilevare l'eventuale collisione prima di aver trasmesso completamente il pacchetto più corto. Tanto più è lungo, in metri, il pacchetto di lunghezza minima, tanto più potranno essere distanti le due stazioni alla estremità della rete. Al crescere della velocità di trasmissione e a parità di dimensione in byte, la lunghezza in metri del pacchetto più corto diminuisce; questo significa che si dovrà ridurre opportunamente la distanza tra le due stazioni. Se a 10 Mb/s un pacchetto di 64 byte avrà una lunghezza di x metri, a 100 Mb/s sarà lungo x/10 metri, a 1 Gb/s sarà lungo x/100 metri; il diametro della rete verrà quindi ridotto di conseguenza. Poiché l'estensione massima di una rete Ethernet a 10 Mb/s è di circa 2 km, a 100 Mb/s tale estensione è ridotta a circa 200 m e a 1 Gb/s a circa 20 m. Al contrario se si sceglie di aumentare la lunghezza minima del pacchetto, mantenendo fisse le dimensioni della rete, si avrà:

Velocità	Dimensione del pacchetto
10 Mb/s	64 byte
100 Mb/s	640 byte
1 Gb/s	6.4 Kbyte

Nei paragrafi seguenti vedremo le scelte effettuate per Fast Ethernet (operante a 100 Mb/s) e per Gigabit Ethernet (operante a 1 Gb/s).

7.6 Fast Ethernet

Esistono due standard per Ethernet a 100 Mb/s:

- **802.3u** o **100 Base T**: nata da una proposta 3Com-Synoptics, mantiene l'originale protocollo CSMA/CD;
- **802.12** o **100VG Anylan**: nata da una proposta HP-AT&T, sfrutta una nuova tecnologia di accesso basata su un modello detto Demand Priority.

7.6.1 802.3u o 100BaseT

È l'unica LAN che possa definirsi Ethernet a 100 Mb/s, perché mantiene inalterato l'originale protocollo CSMA/CD, implementato su doppino o su fibra ottica a 100 Mb/s. La dimensione minima e il formato della

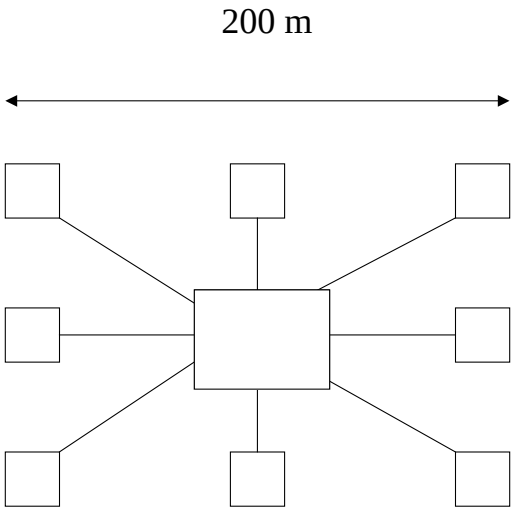


Figura 7.7. Ethernet a 100 Mb/s

trama non sono stati alterati rispetto allo standard 802.3; di conseguenza si sono dovute diminuire le distanze, riducendo la rete a 200 metri di diametro massimo (si veda la figura 7.7). Si adotta una topologia stellare e sono supportate le seguenti modalità operative:

- **half duplex**: quando si usa come centrostella un hub; in tal caso si ha banda condivisa;
- **full duplex**: per connettere tra loro due switch o uno switch e una stazione, quindi con banda dedicata.

La trasmissione a livello fisico utilizza una codifica 8B6T che trasforma un ottetto binario in 6 simboli ternari; lo schema di codifica definisce 256 parole di codice basate su simboli ternari "+", "−" e "0", equivalenti ai 256 valori rappresentabili su 8 bit. La tabella di codifica è costruita in modo tale da garantire un numero di transizioni sufficiente al mantenimento della sincronizzazione del ricevitore con il trasmettitore. Ricordiamo in proposito che sequenze troppo lunghe di 0 e 1 rendono difficile l'aggancio del segnale da parte dei PLL che costituiscono l'elettronica di ricezione delle stazioni.

7.6.2 802.12 o 100BaseVG

Combina la trasmissione di pacchetti Ethernet e Token Ring. È mantenuto solo il formato della trama, mentre il MAC a collisione è sostituito con un **MAC Demand Priority Access Method** (DPAM). Questo protocollo garantisce a tutte le stazioni una minima velocità trasmissiva; la trasmissione è in effetti a 100 Mb/s, ma ogni stazione può trasmettere solo quando abilitata. È inoltre garantito un intervallo massimo tra la richiesta di trasmissione e l'abilitazione a eseguirla; in questo modo si assicura l'arrivo un pacchetto al destinatario entro un tempo massimo. Tutto ciò rende questa tecnologia adatta ad applicazioni multimediali. Il mezzo fisico è costituito da 4 canali che funzionano in modalità half duplex. Su ogni canale la trasmissione avviene a 25 Mb/s con una codifica NRZ. Tale codifica consiste nell'associare a ciascun bit un valore stabile per la sua intera durata, in modo equivalente quindi alla rappresentazione in termini di 0 e 1 (si veda la figura 7.8). Nel

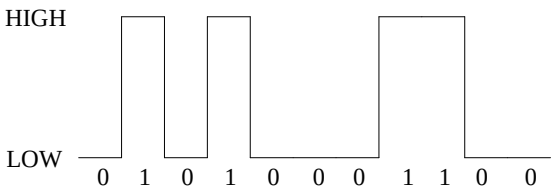


Figura 7.8. Codifica NRZ

caso della configurazione NRZ, una sequenza di valori uguali non genera alcuna transizione, pertanto risulta impossibile la corretta sincronizzazione. Questo problema è aggirato allungando e ricodificando le sequenze di bit da trasmettere in modo da garantire sempre un numero minimo di transizioni. Quando una stazione vuole accedere al mezzo trasmissivo avanza una richiesta allo switch. Il 100VG Anylan gestisce due possibili livelli di priorità: normale o alta. Alle richieste ad alta priorità viene garantito l'accesso alla rete prima di quelle a priorità normale. La gestione delle richieste è effettuata dagli switch mediante una procedura di arbitraggio round robin:

- si osservano, ciclicamente e secondo un ordine predefinito, le porte dello switch, in modo da individuare le richieste di trasmissione;
- le richieste sono soddisfatte nello stesso ordine, partendo però da quelle ad alta priorità.

Lo switch ha una lista per ogni livello di priorità; fino a quando non arriva una richiesta ad alta priorità sono servite le richieste a bassa priorità, nell'ordine delle porte da cui provengono. Una richiesta ad alta priorità è servita immediatamente dopo il termine della trasmissione del pacchetto corrente. Prima che lo switch torni a servire la lista a priorità normale, saranno serviti tutti i pacchetti ad alta priorità. Per evitare il blocco delle richieste a bassa priorità, o *"starvation"*, in caso di eccesso di traffico ad alta priorità, lo switch controllerà continuamente i tempi massimi di attesa concordati con i nodi. Se il ritardo di trasmissione supera il tempo prestabilito, lo switch innalzerà immediatamente la priorità delle richieste da bassa a alta.

7.7 Gigabit Ethernet

È stata standardizzata con un addendum all'IEEE 802.3, detto IEEE 802.3z, ed offre compatibilità completa con lo standard originario. Come l'IEEE 802.3u, rappresenta una evoluzione di Ethernet e come tale offre i vantaggi della semplicità del metodo di accesso, sempre CSMA/CD, e dell'alta scalabilità tra le diverse velocità di trasmissione. Si è mantenuta la possibilità di avere a livello fisico sia fibra ottica, sia cavo in rame; la topologia è stellare e la modalità operativa può essere half o full duplex. Il primo problema che si è dovuto affrontare passando dai 100 Mb/s di Fast Ethernet al Gb/s di Gigabit Ethernet è stato l'estensione massima della rete che, mantenendo inalterata la dimensione minima del pacchetto, si sarebbe ridotta a soli 20 m. Volendo portare tale estensione fino a 200 m e garantendo che un'eventuale collisione sia comunque rilevata in tempi utili, è necessario aumentare la dimensione minima dei pacchetti. Si accoda alla trama un'estensione costituita da particolari simboli, che ne garantiscano una lunghezza pari ad almeno 512 byte.

Inoltre è ogni stazione ha la possibilità di trasmettere più pacchetti senza lasciare il mezzo trasmissivo, fino ad un massimo di 8192 byte; questa modalità è detta burst mode. Una stazione che trasmette in burst mode deve estendere, qualora la lunghezza sia minore della minima consentita, solo il primo pacchetto: i seguenti non richiedono estensione. Tra due pacchetti successivi sono inoltre trasmesse sequenze di bit che permettono il mantenimento del sincronismo tra stazioni mittente e destinataria. La codifica trasmissiva usata è 8B10B, che consiste nella codifica di un simbolo composto da 8 bit in uno da 10 bit. La codifica è fatta in modo da garantire un numero di transizioni sufficienti a consentire la sincronizzazione dei PLL del ricevitore.

7.8 Il futuro

Attualmente le reti Ethernet sono state standardizzate solo fino a 1 Gb/s, ma i costruttori si stanno predisponendo per 10 Gbit Ethernet. Le caratteristiche saranno:

- mezzo trasmissivo in fibra ottica, non in rame;
- centrostella costituito solo da switch o router, non da hub;
- tecnologia esclusivamente full duplex.

La tabella seguente vuole rappresentare un paragone fra gli indici di prestazione delle reti CSMA/CD. Come riferimento è stato preso l'indice di prestazione di una Ethernet classica, posto per convenzione uguale a 1.

Tipologia	Velocità	Indice di prestazione
Shared Ethernet	10 Mb/s	1
Switched Ethernet	10 Mb/s	10
Switched Ethernet	100 Mb/s	100
Switched Ethernet	1 Gb/s	1000
Switched Ethernet	10 Gb/s	10000

8

Livello rete

Il livello rete, o livello 3, gestisce il trasferimento di informazioni tra il nodo di partenza e quello di arrivo; opera quindi su tutta la sottorete di comunicazione e non più sul canale tra due nodi adiacenti come il livello 2.

Questo livello è significativo in particolare per reti terrestri a grande distanza con canali punto–punto, mentre per altri tipi di rete (LAN, MAN e in genere reti broadcast) mittente e destinatario sono sempre collegati a due nodi adiacenti e le funzionalità di livello 2 e di livello 3 tendono a confondersi.

Il livello rete gestisce, mediante opportune primitive, le funzioni di:

- *instradamento*: ricerca del percorso da far seguire alle PDU dal nodo di partenza a quello di arrivo; può essere calcolato PDU per PDU (nel caso di reti datagram), oppure connessione per connessione (nel caso di reti connection–oriented);
- *controllo di congestione*: procedure per evitare la congestione della rete, cioè tecniche per l'allocazione delle risorse (prevalentemente buffer) che cercano di garantire un trasferimento efficiente delle PDU;
- *tariffazione*: non è presente in tutti i tipi di reti; mentre in una LAN questa funzione non è necessaria, perché la rete è privata, in una rete via satellite il livello 3 si occupa quasi esclusivamente di questo aspetto.

8.1 Primitive

La specifica del servizio di livello rete definisce quattro gruppi di primitive:

1. N_CONNECT per aprire una connessione,
2. N_DATA, N_DATA_ACKNOWLEDGE e N_EXPEDITED_DATA per lo scambio dei dati,
3. N_RESET per resettare la connessione,
4. N_DISCONNECT per chiudere la connessione.

La sequenza temporale con cui si susseguono le primitive è illustrata nella figura 8.1. In essa è mostrata l'apertura di una connessione, ma il comportamento è analogo per tutte le primitive.

Per l'apertura delle (3)connessioni si usano le primitive:

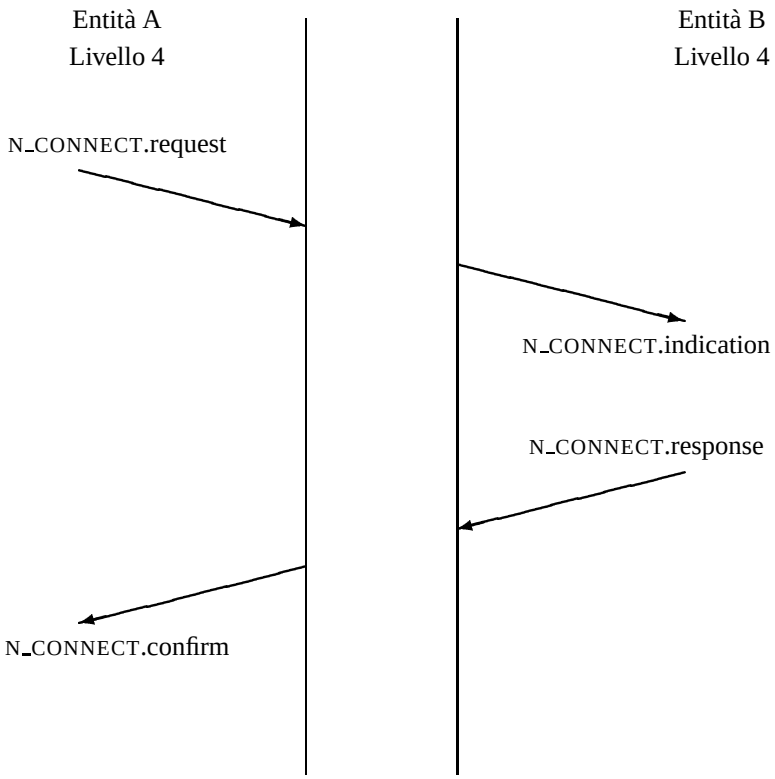


Figura 8.1. Esempio di concatenazione di primitive

- N_CONNECT.request (*da, sa, rcs, eds, qos, data*)
- N_CONNECT.indication (*da, sa, rcs, eds, qos, data*)
- N_CONNECT.response (*ra, rcs, eds, qos, data*)
- N_CONNECT.confirm (*ra, rcs, eds, qos, data*)

con i parametri

- *da* (Destination Address) (3)indirizzo del destinatario (ovvero identificatore del (3)SAP a cui è collegata la (4)entità destinataria).
- *sa* (Source Address) (3)indirizzo del chiamante.
- *rcs* (Receipt Confirmation Selection) specifica se la conferma deve essere locale o remota.
- *eds* (Expedited Data Selection) specifica se si desiderano utilizzare le primitive per la trasmissione di dati expedited.
- *qos* (Quality Of Service parameter set) specifica la qualità di servizio della (3)connessione (velocità, ritardo, probabilità di fallimento, . . .).

- *data* eventuali dati utente.
- *ra* (Responding address) (3)indirizzo del chiamato.

La richiesta di apertura di una (3)connessione può essere rifiutata dalla rete o dal destinatario (cioè dalla (4)entità collegata al (3)SAP di destinazione), attivando una primitiva di tipo N_DISCONNECT; nella figura 8.2 è mostrato il rifiuto da parte della (4)entità chiamata mediante una N_DISCONNECT.request.

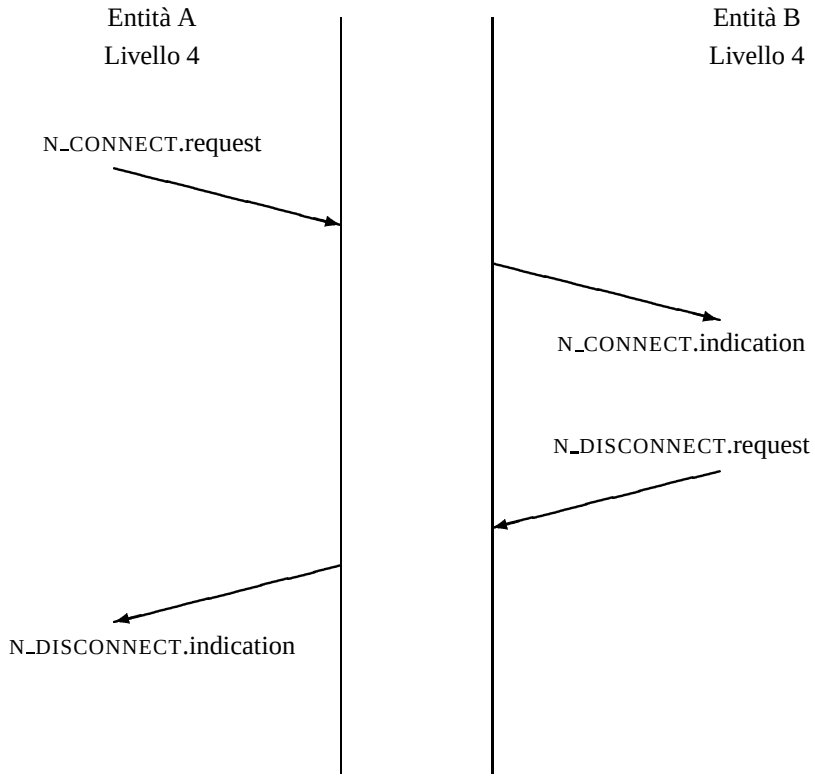


Figura 8.2. Richiesta di connessione rifiutata dal chiamato

Per lo scambio dei dati si usano le primitive:

- N_DATA.request (*data, conf*)
- N_DATA.indication (*data, conf*)
- N_DATA_ACKNOWLEDGE.request ()
- N_DATA_ACKNOWLEDGE.indication ()
- N_EXPEDITED_DATA.request (*data*)
- N_EXPEDITED_DATA.indication (*data*)

con i parametri:

- *data* (3)SDU da trasferire.
- *conf* richiesta di conferma.

Il trasferimento delle (3)SDU avviene generalmente senza conferma remota, come mostrato nella figura 8.3, in quanto il livello rete non si occupa della gestione degli errori. Qualora sia desiderabile, è però possibile chiedere la conferma remota dando un valore opportuno al parametro *conf* (figura 8.4).

Le PDU di tipo EXPEDITED hanno priorità, nelle code, rispetto alle PDU normali e agli ACK. Queste PDU sono generalmente utilizzate per scambio di messaggi di controllo e non hanno ACK.

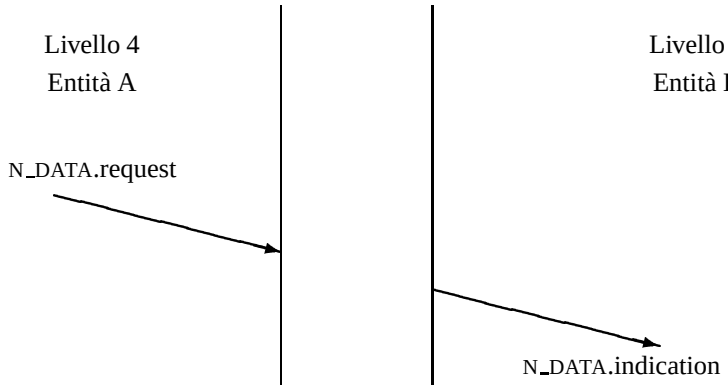


Figura 8.3. Scambio di PDU senza ACK

Per la chiusura della connessione si usano le primitive:

- N_DISCONNECT.request (*rsn, data, ra*)
- N_DISCONNECT.indication (*org, rsn, data, ra*)

con i parametri:

- *rsn* (ReaSoN) Motivo della richiesta.
- *data* Eventuali dati utente.
- *ra* (Responding Address) (3)indirizzo del destinatario.
- *org* (ORiGinator) Identificativo di chi ha richiesto la chiusura.

La chiusura di una connessione può essere richiesta da un utente (una (4)entità) o dal gestore della rete per problemi interni (tipicamente per congestione). In questo caso i due utenti ricevono la primitiva N_DISCONNECT.indication. Si noti che la chiusura di una (3)connessione è brusca, quindi con possibile perdita di PDU in transito.

Qualora non sia più possibile la comunicazione si può effettuare un “reset” della (3)connessione, che è un’operazione equivalente alla chiusura e alla successiva riapertura della (3)connessione stessa. Le primitive che consentono questa operazione sono:

- N_RESET.request (*rsn*)

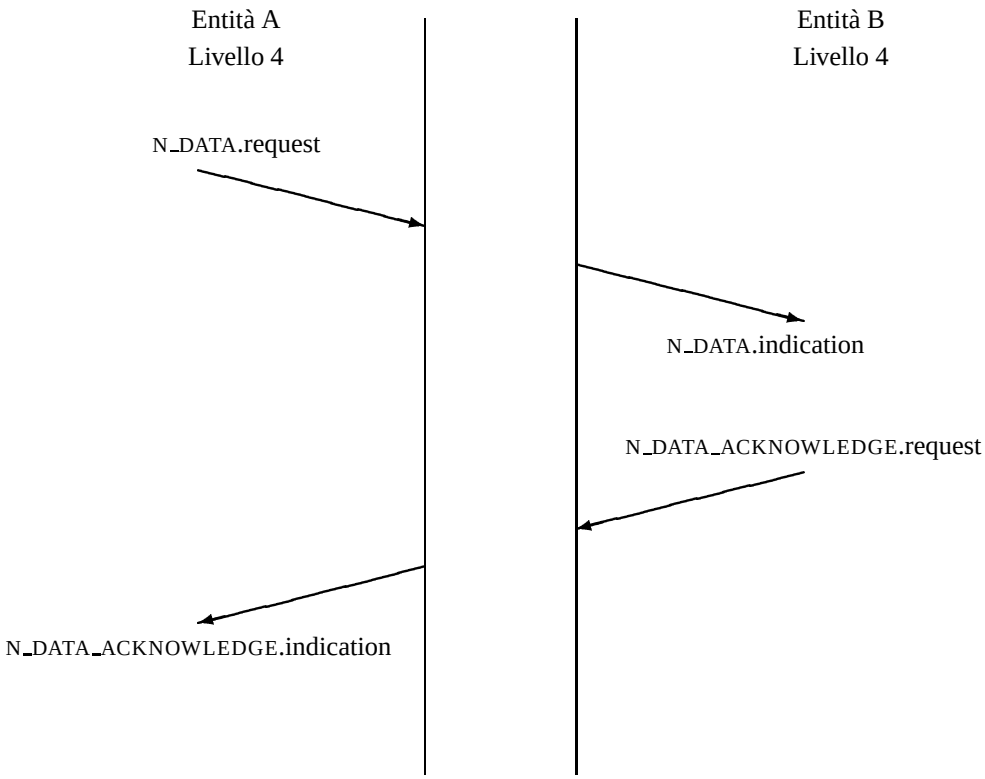


Figura 8.4. Scambio di PDU con ACK

- N_RESET.indication (*org*, *rsn*)
- N_RESET.response ()
- N_RESET.confirm ()

con i parametri:

- *rsn* (ReaSoN) Motivo del reset.
- *org* (ORiGinator) Identificativo di chi ha causato il reset (può anche essere il gestore di rete).

8.2 Tecniche di instradamento

La funzione dell'instradamento ha una diversa rilevanza in reti datagram e reti a circuito virtuale.

In reti a circuito virtuale, all'apertura della (3)connessione tra due (3)SAP che servono due (4)entità, si stabilisce un percorso lungo il quale sono inviate tutte le (3)PDU, che seguono quindi la stessa strada. La procedura di instradamento viene eseguita solo una volta per tutte le PDU inviate sulla connessione.

In reti datagram le (3)PDU possono seguire strade diverse poiché la procedura di instradamento viene eseguita per ogni PDU.

È evidente che l'instradamento risulta meno oneroso per i servizi a circuito virtuale rispetto ai servizi datagram. Supponiamo di dover inviare P PDU, che attraversano N nodi; utilizzando un servizio di tipo circuito virtuale è necessario effettuare l'instradamento una sola volta all'atto dell'apertura della (3)connessione, quindi si deve decidere il percorso N volte, cioè una volta per nodo. Invece, utilizzando un servizio di tipo datagram bisogna scegliere il percorso NP volte, cioè in ogni nodo e per ogni PDU inviata.

È inoltre ovvio che la funzione dell'instradamento è importante in reti con topologia a maglia non completamente connessa, mentre è irrilevante (inutile) per reti broadcast.

Compito dell'algoritmo di instradamento in ogni nodo è riconoscere se le PDU ricevute sono destinate ad utenti collegati al nodo stesso (ed in tal caso estrarle dalla rete e consegnarle all'utente destinatario) o, in caso contrario, decidere su quale portante fisico uscente dal nodo ritrasmetterle (instradarle).

Le tecniche di instradamento possono essere classificate in due categorie fondamentali:

- instradamento adattativo
- instradamento non adattativo

a seconda che siano in grado o meno di adattarsi alle variazioni di traffico nella rete.

Per ogni categoria esistono molte tecniche diverse, per esempio:

- instradamento non adattativo
  - instradamento a inondazione (flooding)
  - instradamento statico
- instradamento adattativo
  - instradamento centralizzato
  - instradamento isolato
  - instradamento distribuito

I requisiti di una politica di instradamento si possono riassumere come segue.

correttezza: le informazioni devono arrivare effettivamente al (3)SAP destinatario;

semplicità: gli algoritmi devono essere semplici, in modo da non richiedere molte risorse di calcolo e/o memoria per la loro realizzazione;

robustezza: in modo da essere resistenti ai guasti ed alle variazioni di ambiente (dall'inglese *robust)*;

ottimalità: cioè deve utilizzare le risorse disponibili nella rete nel modo migliore;

equità: gli utenti devono essere trattati tutti nello stesso modo.

8.2.1 Instradamento ad inondazione

L'algoritmo di instradamento a inondazione (o flooding) prevede che una PDU entrante in un qualsiasi nodo della rete (ad esclusione del nodo destinatario) da un certo canale sia inoltrata su tutti i canali in uscita da quel nodo, tranne che sul canale di provenienza. Quando la PDU arriva a destinazione non viene più replicata.

Questa tecnica è detta instradamento ad inondazione, perché la rete viene inondata di PDU; le PDU sono eliminate solo quando raggiungono la destinazione.

È una tecnica molto semplice da implementare, ma molto onerosa, che viene utilizzata soprattutto in reti di topologia ignota e molto soggetta a variazioni, tali per cui sia molto difficile ottenere le informazioni necessarie per realizzare una tecnica di instradamento più raffinata. Un esempio di applicazione si trova nelle reti militari, in cui è necessaria la certezza di ricevere le informazioni strategiche anche in condizioni ambientali molto critiche (interruzioni di collegamenti a causa di azioni ostili, ecc.).

Questa tecnica presenta il vantaggio che, poiché vengono percorse tutte le strade che collegano il mittente al destinatario, sarà scelta anche la più breve e la meno intasata. In questo modo, senza conoscere la topologia, con una decisione puramente locale e indipendente da ciò che accade nella rete, viene scelta sicuramente anche la soluzione ottima.

Lo svantaggio evidente è che si produce nella rete una enorme quantità di traffico. È possibile introdurre alcuni correttivi, utilizzando ad esempio:

- un indicatore (un bit per ogni nodo), che segnala se la PDU è già transitata una volta per ogni nodo. Al secondo passaggio la PDU non viene più propagata. In questo modo si limita il numero di PDU che circolano nella rete: si introduce nella rete un picco di traffico, che poi si esaurisce;
- un contatore di duplicazione, che viene incrementato ad ogni passaggio in un nodo. Quando il numero di passaggi raggiunge un valore di soglia prefissato, la PDU viene scartata. Il valore di soglia può essere pari al numero di nodi nella rete oppure alla lunghezza del percorso massimo tra due nodi della rete; in questo secondo caso è necessario però possedere informazioni sulla topologia della rete.

Oltre che in applicazioni di tipo militare, questa tecnica è utilizzata in sistemi per interconnessione di LAN.

8.2.2 Instradamento statico

In una rete con topologia magliata, in ogni nodo si deve decidere quale tra i nodi adiacenti è il più adatto per permettere alla PDU di raggiungere la propria destinazione. La decisione viene presa con l'aiuto di una tabella residente nel nodo, che contiene due (o più) colonne:

- nella prima colonna sono contenuti i nomi di tutti i possibili destinatari (tutti i (3)SAP);
- nella seconda colonna, per ogni (3)SAP è indicato il nome del nodo adiacente (e quindi l'identificatore del portante fisico) a cui indirizzare la PDU perché raggiunga la destinazione finale richiesta.

Per esempio nel caso della figura 8.5, se la PDU si trova nel nodo I, si deve decidere a quale tra i nodi adiacenti inviarla affinché possa raggiungere il nodo Z. La tabella 8.1 contiene le informazioni necessarie per l'instradamento.

La tabella è creata e aggiornata dalle funzioni di gestione della rete; gli aggiornamenti avvengono in modo molto saltuario, per seguire le evoluzioni della topologia della rete. Non vi è invece alcun intervento in funzione delle variazioni di traffico.

Questo tipo di instradamento è detto *statico non suddiviso*, perché in un certo nodo tutte le PDU dirette alla stessa destinazione finale vengono inoltrate in modo deterministico sempre allo stesso nodo adiacente.

L'instradamento *statico suddiviso* permette di offrire due (o più) scelte possibili per la stessa destinazione finale: la tabella in questo caso, oltre all'indicazione delle due scelte possibili, conterrà la percentuale del traffico da inviare su un canale oppure sull'altro (la somma del traffico inviato su tutti i canali dovrà ovviamente essere pari a 1 – si veda la tabella 8.2).

Se si guasta un canale, il gestore della rete interviene e modifica i valori delle percentuali di traffico da inoltrare sul nodo interessato dal guasto.

Tecniche simili sono molto utilizzate anche nelle reti a commutazione di circuito.

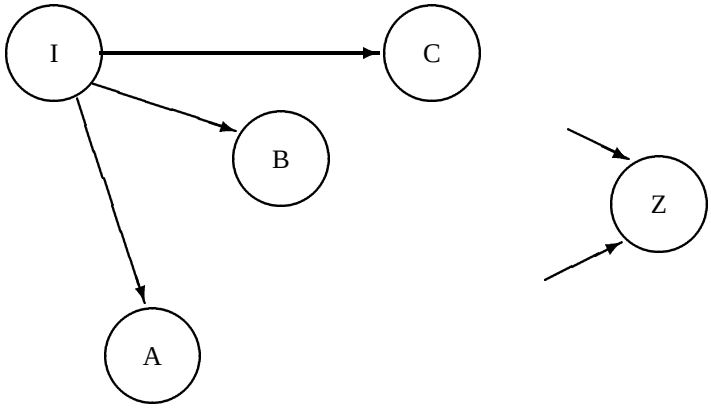


Figura 8.5. Esempio di rete

Destinatari	Nodo adiacente
A	A
B	B
C	C
D	A
⋮	⋮
Z	B

Tabella 8.1. Tabella per la tecnica di instradamento statico non suddiviso

8.2.3 Instradamento centralizzato

È l'algoritmo più semplice di instradamento adattativo. Nella rete normalmente esiste un centro di controllo (Network Control Center – NCC) che gestisce la raccolta di dati dalla periferia e controlla che la rete funzioni

Destinatari	Prima scelta	%	Seconda scelta	%
A	A	0.8	B	0.2
B	B	0.7	C	0.3
C	C	0.1	D	0.9
D	A	0.4	B	0.6
⋮	⋮	⋮	⋮	
Z	B	0.8	Q	0.2

Tabella 8.2. Tabella per la tecnica di instradamento statico suddiviso

in modo corretto. Questo NCC può farsi carico anche dell'instradamento, poiché costituisce già un punto di raccolta delle informazioni su guasti, situazioni di congestione, ecc. La frequenza di aggiornamento delle tabelle diviene tale da permettere di intervenire anche sulle fluttuazioni del traffico.

Il funzionamento non è qualitativamente diverso da quello dell'instradamento di tipo statico, varia solo la frequenza con cui sono aggiornate le tabelle. La differenza è riconducibile a quella tra:

- gestione: si segue la dinamica delle eccezioni, degli interventi rari;
- controllo: si interviene con una frequenza maggiore, cercando di seguire le fluttuazioni del traffico.

La differenza è quindi molto labile, legata solo al valore di costanti di tempo.

Il NCC:

1. raccoglie periodicamente informazioni da tutti i nodi della rete sul loro stato;
2. elabora le nuove tabelle di instradamento per tutti i nodi della rete, tenendo conto delle nuove informazioni che ha ricevuto;
3. spedisce le nuove tabelle ai nodi.

Questo tipo di soluzione presenta alcuni svantaggi.

- La funzione di instradamento è affidata ad un solo elemento, quindi è particolarmente vulnerabile ai guasti del NCC. Se il NCC si guasta, l'instradamento continua ad essere eseguito con l'ultimo insieme di tabelle che è stato distribuito. Se si verifica un malfunzionamento per cui il NCC distribuisce tabelle sbagliate, la situazione è ancora peggiore. La soluzione consiste nell'uso di NCC progettati in modo da avere un'affidabilità elevata ed eventualmente nella loro duplicazione.
- Questo tipo di funzionamento è strutturalmente poco equilibrato: esistono momenti in cui è presente un volume elevato di traffico entrante nel NCC (raccolta di informazioni), altri in cui è elevato il volume di traffico in uscita dal NCC (spedizione delle tabelle ai nodi periferici). Esistono quindi pulsazioni di traffico con periodicità data dalla frequenza di aggiornamento delle tabelle (ordine di grandezza del minuto). Il flusso di traffico è significativo e può creare congestione, quindi rende necessario il sovradimensionamento della rete.
- Le nuove tabelle devono essere distribuite a tutti i nodi: poiché è possibile che nodi diversi ricevano la nuova tabella in tempi diversi, esiste un problema di consistenza delle tabelle. Ogni versione delle tabelle è consistente al suo interno, ma non è possibile imporre vincoli di consistenza tra versioni diverse delle tabelle. Bisogna sincronizzare l'uso delle nuove tabelle: le tabelle contengono l'indicazione del tempo t_0 della loro abilitazione; in questo caso però è necessario che tutti gli orologi nella rete siano sincronizzati.

L'instradamento centralizzato, a fronte di questi svantaggi, presenta il vantaggio di un algoritmo estremamente semplice, perché il NCC dispone localmente di tutte le informazioni necessarie ed è possibile reagire in modo immediato a qualsiasi fenomeno che si verifichi nella rete.

8.2.4 Instradamento isolato

È una tecnica di instradamento adattativo, che utilizza tecniche di ottimizzazione locale, in cui ciascun nodo utilizza l'informazione localmente disponibile per calcolare gli instradamenti. I nodi non si parlano tra loro e non si cerca di ricostruire un'immagine locale dello stato globale della rete.

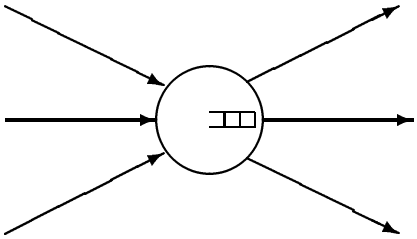


Figura 8.6. Modello di un nodo che realizza una tecnica di instradamento isolato

Il caso ottimo di confronto è la tecnica di instradamento centralizzato, utilizzando la tabella che sarebbe calcolata da un NCC che disponga di tutte le informazioni sullo stato della rete.

Il nodo rappresentato nella figura 8.6 è in grado di valutare la situazione delle code in uscita ed ha la possibilità di svolgere elaborazioni sulle PDU in transito. Esistono diverse strategie possibili.

Instradamento 'hot potato' e a deflessione

L'informazione relativa alle code in uscita può essere sfruttata inviando la PDU all'uscita dove esiste la coda più breve, minimizzando il tempo di permanenza della PDU all'interno del nodo. In questo caso non si tiene conto della destinazione della PDU. Il metodo è detto *hot potato* perché ogni nodo cerca di liberarsi della PDU il più rapidamente possibile.

A questo tipo di instradamento si può sovrapporre una gestione a tabella: tra un certo numero di scelte indicate nella tabella, si sceglie in base alla lunghezza delle code sulle uscite, anziché in termini probabilistici.

Questa tecnica presenta alcuni vantaggi:

- esigenze di elaborazione ridotte;
- occupazione omogenea dei canali in uscita.

Lo svantaggio è che non è possibile determinare se e quando la PDU arriverà a destinazione.

Per alcune topologie di rete questa tecnica di instradamento è quella preferita: sono reti molto magliate, con topologie regolari, quali per esempio la rete a maglia quadrata con topologia toroidale (detta anche Manhattan) rappresentata nella figura 8.7.

In una rete di questo tipo esiste un gran numero di percorsi possibili tra il nodo mittente ed il destinatario. Consideriamo il modello semplificato di nodo rappresentato nella figura 8.8: i canali sono unidirezionali ed ogni nodo ha due ingressi e due uscite. Utilizzando la tecnica hot potato, tutte le PDU che entrano sono instradate o verso il basso o verso destra, a seconda del traffico. Esistono alcuni casi particolari che vale la pena di valutare: per esempio, se il nodo si trova sulla verticale del nodo destinatario, se la PDU è instradata verso destra dovrà fare un percorso più lungo. Si aggiunge allora una tabella, che permette di vincolare le scelte in questi casi particolari e di raggiungere una buona efficienza di funzionamento.

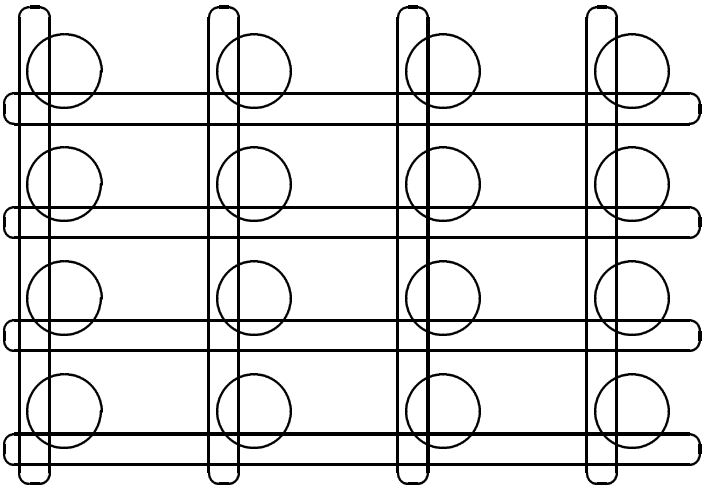


Figura 8.7. Rete con topologia Manhattan

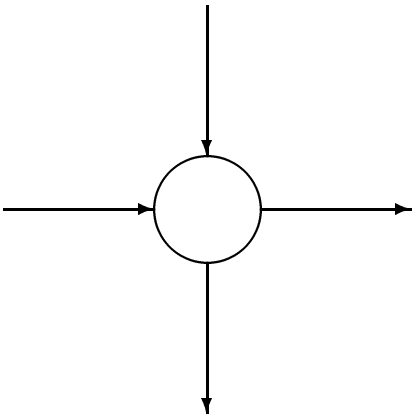


Figura 8.8. Modello semplificato di un nodo

Esiste una variante di questo metodo in cui il funzionamento dei nodi è sincronizzato e le PDU hanno tutte la stessa lunghezza. In questo caso, ad ogni intervallo di tempo, detto *slot*, in ogni nodo possono verificarsi le seguenti situazioni:

- non entra nessuna PDU;
- entra una sola PDU, che può essere instradata sull'uscita appropriata;
- entrano due PDU: se entrambi richiedono di essere instradate sulla stessa uscita, vengono accodati uno

dopo l'altro. È necessaria una memoria per contenere le PDU in coda sulle uscite.

Esiste una versione semplificata di questo algoritmo, detta instradamento *a deflessione* in cui il nodo è quasi privo di memoria: quando due PDU richiedono la stessa uscita, una è instradata nella direzione richiesta e l'altra viene deflessa, cioè inviata in una direzione diversa da quella richiesta. Il fatto che la rete sia molto magliata fa sì che le PDU deflesse possano comunque arrivare a destinazione, pur seguendo un percorso più lungo.

Instradamento backward learning

Un altro algoritmo di instradamento isolato è quello detto di *backward learning*. Il nodo sceglie il canale su cui instradare ogni PDU in base al contenuto delle tabelle dei costi di ciascuna uscita; a partire da una stima affidabile del costo, si sceglie il canale a costo minore.

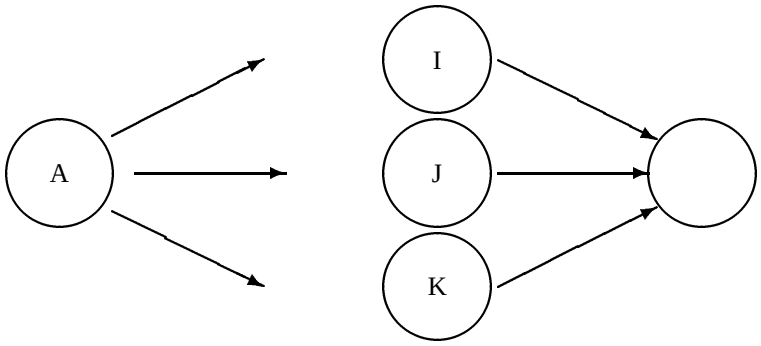


Figura 8.9. Possibili percorsi tra due nodi

Per eseguire la stima dei costi, è possibile sfruttare l'informazione portata implicitamente al nodo dalle PDU in arrivo (figura 8.9). Analizziamo, per esempio, le vicende delle PDU partite da A ed arrivate al nodo attraverso I, J e K: se la PDU proveniente da I è stata molto più veloce, si suppone che la strada migliore per raggiungere A passi dal nodo I. Questa conclusione è basata sulla ipotesi di traffico isotropo: si suppone cioè che il traffico e le dimensioni delle code siano omogenei in entrambi i versi di percorrenza del collegamento tra due nodi.

Esiste un campo nell'intestazione della PDU che contiene un indicatore di qualità del percorso utilizzato per raggiungere il nodo in esame. Questo indicatore può essere:

- il tempo di generazione: posso calcolare il ritardo della PDU, ma è necessario che gli orologi della rete siano sincronizzati;

- un contatore di nodi attraversati: è meno efficace, ma non richiede alcun tipo di sincronizzazione.

8.2.5 Instradamento distribuito

In questo caso i nodi vicini (collegati con un canale punto–punto) possono comunicare tra loro e scambiarsi informazioni utili per costruire una rappresentazione locale dello stato globale della rete.

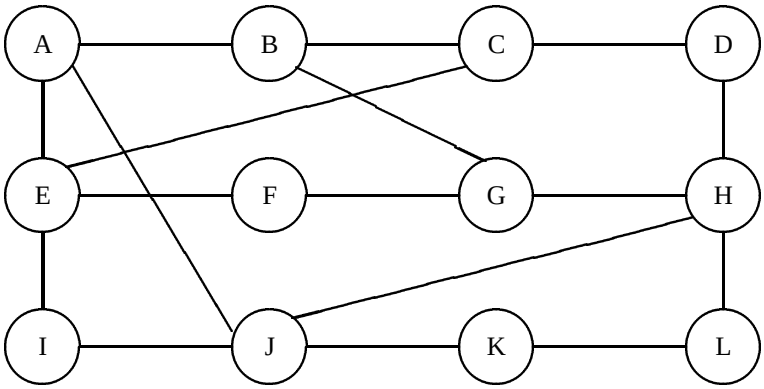


Figura 8.10. Rete magliata

Consideriamo, per esempio, la rete magliata della figura 8.10 e analizziamo il comportamento del nodo J. Il nodo J, adiacente ai quattro nodi A, H, I e K, è in grado di stimare localmente il costo dei percorsi verso gli altri quattro nodi in termini di lunghezza delle code interne e di tempo di propagazione nelle direzioni considerate.

	da A	**da H**	**da I**	**da K**
A	0	20	24	21
B	12	31	36	28
C	25	19	18	36
D	40	8	27	24
E	14	7	30	22
⋮	⋮	⋮	⋮	⋮
K	16	30	7	0
L	29	9	33	9

Tabella 8.3. Nodo J: valori di costo ricevuti dai nodi limitrofi

Il nodo J riceve periodicamente dai nodi limitrofi la loro valutazione dei costi per raggiungere ogni nodo della rete (una colonna della tabella di instradamento) e ricalcola la propria tabella in base alle informazioni che ha ricevuto. Se per esempio il nodo J riceve i costi riportati nella tabella 8.3, calcola una nuova tabella di costi sommando a ciascun valore ricevuto il costo aggiuntivo per andare da se stesso al vicino (supponiamo 8 per andare ad A, 12 per andare a H, 10 per andare ad I e 6 per andare a K) e ottiene la tabella 8.4.

Il nodo J sceglie poi il percorso a costo minimo verso ciascun nodo e genera una nuova tabella interna (tabella 8.5), la cui colonna dei costi sarà poi inviata a tutti i nodi limitrofi.

	A	H	I	K
A	8	32	34	27
B	20	43	46	34
C	33	31	28	42
D	48	20	37	30
E	22	15	40	28
⋮	⋮	⋮	⋮	⋮
L	37	21	17	15

Tabella 8.4. Tabella dei costi

Destinazione	Costo	Nodo adiacente
A	8	A
B	20	A
C	28	I
D	20	H
⋮	⋮	⋮
K	6	K
L	15	K

Tabella 8.5. Nuova tabella interna del nodo J

Esistono alcuni problemi:

- la consistenza delle tabelle, perché l'aggiornamento non è sincrono. La soluzione è la sincronizzazione di tutti i nodi della rete e la sostituzione contemporanea delle tabelle di tutti i nodi;
- l'instradamento non è costruito su una conoscenza globale della rete, ma solo sulle informazioni disponibili all'interno del nodo e fornite dagli immediati vicini. Se un canale si guasta è necessario un certo numero di iterazioni prima che l'informazione si propaghi all'interno della rete. L'algoritmo reagisce meglio all'inserimento di un nuovo nodo nella rete che ai guasti; ha efficienze buone perché i guasti non sono eventi molto frequenti.

8.3 Controllo di congestione

La congestione si verifica a causa di contese interne ad un nodo per l'allocazione di risorse. Una delle cause di congestione è la disponibilità limitata di memoria all'interno dei nodi: può succedere che tutta la memoria disponibile all'interno di un nodo sia utilizzata per mantenere una coda di PDU in attesa di essere trasmesse. La situazione è rappresentata schematicamente nella figura 8.11; due nodi hanno saturato la memoria disponibile con PDU in attesa di essere inoltrate; è una condizione di *blocco diretto* (o deadlock), in cui per liberare una risorsa in B è necessario liberare una risorsa in A e viceversa. Esistono naturalmente forme più complicate di blocco, causate da collegamenti di insiemi di nodi ad anello.

La congestione nasce quindi da problemi di gestione delle risorse interne di un nodo e può essere risolta:

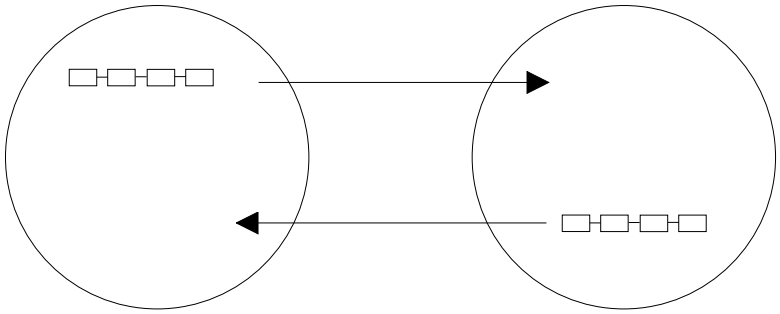


Figura 8.11. Condizione di blocco

- localmente, utilizzando algoritmi per la gestione dei buffer all'interno del nodo;
- globalmente, su tutta la rete.

8.3.1 Gestione locale

Si deve definire una politica di gestione dei buffer all'interno di un nodo.

La soluzione più semplice è la *preallocazione dei buffer*, che sono utilizzati in modo esclusivo da una connessione. Tutte le risorse, ad esclusione della banda del canale, sono allocate staticamente tra due utenti: è una tecnica semplice, che porta però ad una scarsa efficienza nell'uso delle risorse del nodo. La PDU, come nella commutazione di circuito, dopo l'apertura del circuito virtuale trova sempre le risorse disponibili lungo il percorso.

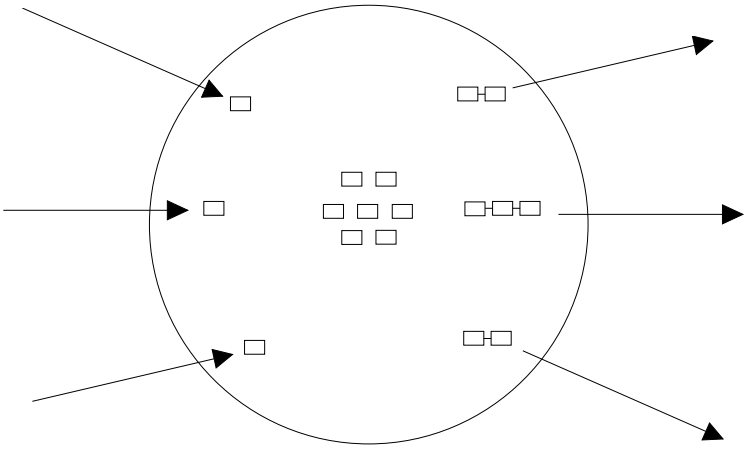


Figura 8.12. Preallocazione parziale delle risorse

In alternativa, è possibile preallocare solo una parte delle risorse, come indicato nella figura 8.12, e

precisamente:

- si assegna un buffer per ogni canale entrante, in modo da poter sempre leggere le PDU in ingresso (potrebbero essere ACK, che permettono di liberare risorse);
- ad ogni coda di uscita sono preallocati in modo statico alcuni buffer;
- esiste un insieme di buffer disponibili, che possono essere allocati ad una coda qualsiasi.

Quando arriva una PDU, viene letta e:

- se c'è posto, viene inserita in una delle code di uscita;
- se non esistono risorse disponibili viene scartata.

La preallocazione sulle code in uscita permette di inoltrare i dati al massimo parallelismo consentito dalla struttura hardware, evitando di allocare tutti i buffer sullo stesso canale (possibilità di blocco). Esistono tecniche di allocazione ottima dell'insieme di buffer che sono associati dinamicamente ai canali d'uscita in funzione del traffico.

Una tecnica alternativa consiste nel dividere la memoria disponibile in un nodo in gruppi di buffer, che possono essere utilizzati da PDU che sono arrivate nel nodo dopo aver attraversato un certo numero di canali.

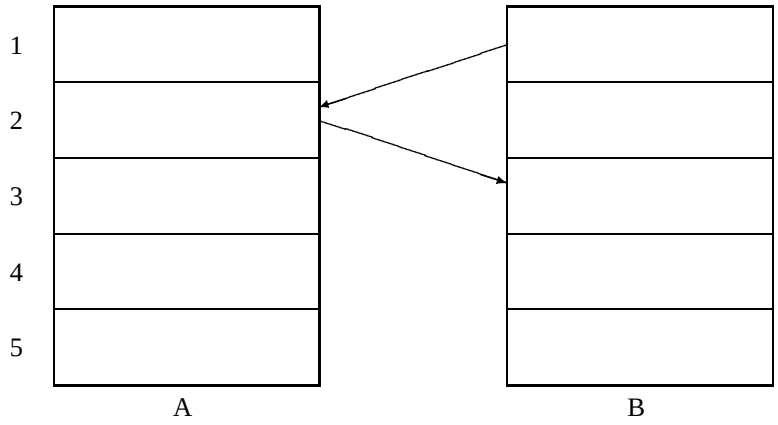


Figura 8.13. Gestione locale a gruppi

Per esempio (si veda la figura 8.13), una PDU arrivata in A dopo aver attraversato 2 nodi, preleva un buffer dal gruppo 2 di A e, quando arriva in B, avrà attraversato 3 nodi, e richiederà risorse al gruppo di buffer 3, che è diverso, evitando così che A aspetti B e B aspetti A sullo stesso gruppo di risorse. In questo modo si evitano tutti i blocchi semplici rendendo asimmetrico il problema: la asimmetria è data dal numero di canali attraversati.

8.3.2 Gestione globale

La tecnica *isaritmica* permette di limitare il fenomeno della congestione a livello di rete, limitando il numero totale di PDU che possono circolare nella rete in un istante qualsiasi.

Si definiscono PDU di controllo, dette *crediti*; la somma totale di PDU dati e crediti all'interno della rete deve essere costante. Quando un utente vuole inserire una PDU nella rete, deve ottenere un credito dal nodo

a cui è collegato: eliminato il credito, è possibile introdurre una PDU nella rete. Se il numero complessivo di PDU dati e crediti è dimensionato in modo corretto, si riduce notevolmente la possibilità di congestione.

Esistono alcuni svantaggi:

- è difficile individuare le condizioni di malfunzionamento in cui un nodo estrae o genera crediti a ripetizione, perché è difficile controllare che il numero totale di PDU dati e crediti rimanga effettivamente costante nella rete;
- possono verificarsi fenomeni di monopolizzazione della rete da parte di due utenti che dialogano intensamente: entrambi gli utenti catturano crediti e li spendono localmente per spedire le loro PDU. Per rimediare a questo problema, si può decidere che i crediti generati localmente, prima di essere riutilizzati debbano essere immessi nella rete. Questa soluzione però penalizza proprio coloro che utilizzano più intensamente la rete.

9

Livello trasporto

Il livello 4 o trasporto è il primo livello che si occupa del trasferimento dell'informazione dall'utente sorgente al destinatario; in altre parole ha caratteristiche end-to-end e non si occupa delle operazioni che coinvolgono la rete al suo interno (i sistemi relay non sono interessati dai livelli superiori al 3).

L'unità dati elementare è del servizio di trasporto è la (4)SDU, detta anche messaggio; essa viene suddivisa in (4)PDU di dimensione compatibile con la massima ammessa per le (3)SDU, dette anche pacchetti. Il compito principale del livello trasporto è quello di fornire un servizio di trasferimento dell'informazione di qualità accettabile, indipendentemente dal tipo di rete su cui si appoggia. Questo compito può essere più o meno facile a seconda della qualità della rete.

9.1 Classi di trasporto

Lo standard OSI prevede cinque classi di protocollo di livello trasporto, chiamate classi 0, 1, 2, 3 e 4. Al crescere del numero che identifica la classe, le funzionalità offerte sono sempre più sofisticate e il protocollo diventa più complicato. La scelta della classe di protocollo è in genere fatta in base alla qualità del servizio fornito dalla rete: se la rete ha caratteristiche di buona qualità, si può utilizzare il protocollo di trasporto di classe 0. I protocolli di classe superiore diventano necessari al peggiorare delle caratteristiche della rete.

I due standard che descrivono il livello trasporto sono l'ISO 8072 che specifica il **servizio** di trasporto e l'ISO 8073 che specifica il **protocollo** di trasporto.

In base alle caratteristiche qualitative di una rete, lo standard 8072 identifica tre tipi di rete, dette reti di tipo A, B e C. La rete di tipo A è caratterizzata da una probabilità residua di errore bassa e da cadute della connessione poco frequenti. La rete di tipo B è caratterizzata da cadute di connessione frequenti, ma la probabilità residua di errore è ancora bassa. La rete di tipo C infine ha una elevata probabilità di errore e cadute di connessione molto frequenti. Si osservi che con il termine probabilità di errore residua si intende la probabiltà di errore dopo avere svolto i controlli previsti dai livelli 2 e 3.

Inoltre si noti che non è previsto un tipo di rete "duale" rispetto al caso B: infatti si presume che se ci sono tanti errori residui allora anche il protocollo di rete abbia malfunzionamenti.

Le classi di protocolli di trasporto sono utilizzabili con i tre tipi di rete descritti secondo quanto specificato nella tabella 9.1.

Le funzionalità delle varie classi non crescono in modo monotono al crescere del numero; la classe 2 è un sovrainsieme della 0, come la 3 lo è della 1. La classe 4 offre l'insieme più completo di funzionalità.

Classe di trasporto	Tipo di rete
0	A
1	B
2	A
3	B
4	C

Tabella 9.1. Relazione tra classe di trasporto e tipo di rete

Classe 0

Il protocollo di classe 0 (denominata **Simple class**) non si preoccupa del controllo di flusso, né del controllo di sequenza, né nel controllo di errore; tutto viene demandato al livello 3. Eventuali errori vengono ribaltati sul protocollo di livello 5. Questo perché si appoggia su un servizio di rete sufficientemente affidabile.

I servizi offerti dal protocollo, sono prevalentemente orientati alla connessione. Il protocollo di classe 0 permette di:

- aprire e chiudere connessioni;
- trasferire dati;
- notificare errori.

Classe 1

Il protocollo di classe 1 (**Basic error recovery class**) permette di:

- aprire e chiudere connessioni;
- trasferire dati;
- notificare errori;
- recuperare fallimenti di connessione a livello rete.

Poiché viene utilizzato con reti di tipo B deve gestire i malfunzionamenti interni alla rete e mascherarli rispetto al servizio offerto. La (4)connessione non cade anche se cade la (3)connessione. Il livello trasporto si occuperà di riaprire una (3)connessione per mascherare il malfunzionamento al livello 5.

Classe 2

Il protocollo di classe 2 (chiamato **Multiplexing class**) permette di:

- aprire e chiudere connessioni;
- trasferire dati;
- notificare errori;
- effettuare della multiplazione di connessipni.

Rispetto al protocollo di classe 0 offre la possibilità di multiplare più (4)connessioni su una unica (3)connessione. In questo caso esiste un controllo di flusso opzionale.

Classe 3

Il protocollo di classe 3 (detta **Error recovery class**) permette di:

- aprire e chiudere connessioni;
- trasferire dati;
- notificare errori;
- effettuare della multiplazione;
- recuperare fallimenti di connessione a livello rete.

Classe 4

Questa classe (denominta **Error detection and recovery class**) ha tutte le funzionalità della classe 3 e inoltre permette il recupero degli errori.

Inoltre si ha disposizione la possibilità dello *splitting*: si possono usare più (3)connessioni per un'unica (4)connessione. Questo può essre necessario nel caso di reti con prestazioni scadenti: usando più (3)connessioni in parallelo si può sperare di migliorare il servizio di livello trasporto.

9.2 Primitive

Le primitive del livello trasporto possono essere raggruppate a seconda delle loro funzioni:

- apertura della (4)connessione
  - T_CONNECT.request (*sa, da, eo, qos, data*)
  - T_CONNECT.indication (*sa, da, eo, qos, data*)
  - T_CONNECT.response (*da, eo, qos, data*)
  - T_CONNECT.confirm (*da, eo, qos, data*)
- chiusura della (4)connessione
  - T_DISCONNECT.request (*sa, da, data*)
  - T_DISCONNECT.indication (*sa, da, reason, data*)
- trasferimento dati
  - T_DATA.request (*data*)
  - T_DATA.indication (*data*)
- trasferimento dati veloce
  - T_EXPEDITED_DATA.request (*data*)
  - T_EXPEDITED_DATA.indication (*data*)

I parametri *sa* e *da* rappresentano gli indirizzi di destinazione e di sorgente, cioè rispettivamente il (4)SAP relativo alla (5)entità che ha generato la request ed il (4)SAP della (5)entità che riceve la indication, *eo* (expedited data option) indica se sono attivabili le primitive di trasferimento veloce dei dati, il parametro *data* rappresenta la (4)SDU. Il parametro *qos* rappresenta un insieme di parametri che determinano la qualità del servizio, tra i quali i più importanti sono la velocità del canale, il ritardo nell'apertura della (4)connessione, la probabilità di fallimento dell'apertura della (4)connessione, il ritardo massimo nel trasferimento delle (4)PDU, la probabilità di errore residua, il tempo massimo prima della ricezione, la probabilità di perdita di PDU, ...

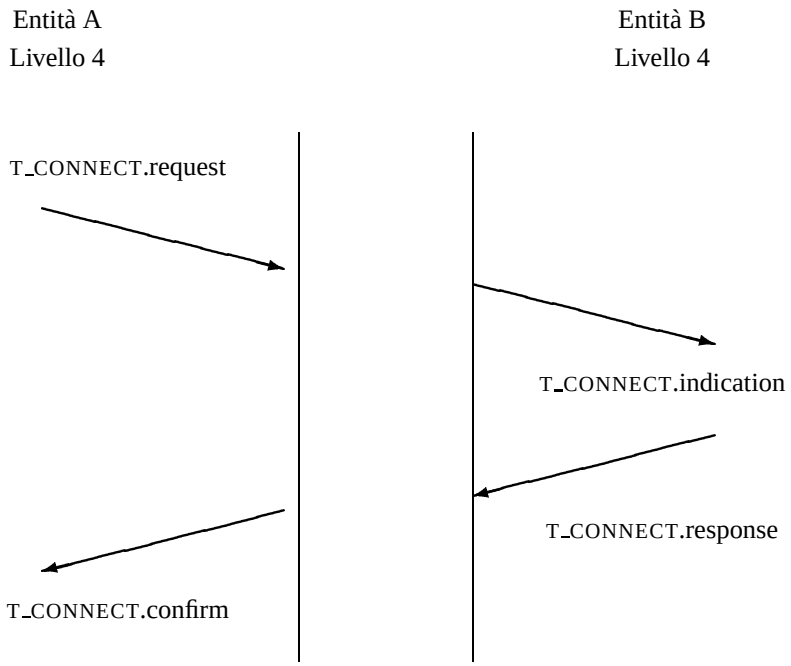


Figura 9.1. Apertura con successo di (4)connessione

Esaminiamo il caso di apertura di una (4)connessione, riportato nella figura 9.1. La (5)entità A chiede l'apertura di una connessione specificando una certa qualità di servizio. Il fornitore del servizio può abbassare la QOS prima di consultare la (5)entità chiamata, se non è in grado di soddisfare la richiesta. La (5)entità chiamata riceve una richiesta di apertura di una (4)connessione con una certa QOS che può ulteriormente ridurre se non è in grado di soddisfarla, ponendo un opportuno valore nel parametro QOS della primitiva di tipo *response*. Il fornitore di servizio riporta questo valore di QOS alla (5)entità chiamante mediante la primitiva di tipo *confirm.* A questo punto la (5)entità chiamante decide se accettare la apertura della (4)connessione con questo valore di QOS o di rifiutarla. Si osservi come non in tutti i parametri che caratterizzano la QOS sono negoziabili, perché alcuni hanno valori fissi non modificabili. Inoltre la QOS che caratterizza la T_CONNECT.confirm deve essere uguale a quella della T_CONNECT.response, altrimenti la (5)entità chiamata non viene messa al corrente dell'ulteriore degrado nei parametri di qualità della (4)connessione. Inoltre il fatto che non tutti i parametri di QOS vengano specificati crea un problema di formato delle (4)PDU, che non possono essere di formato fisso.

Le procedure di chiusura di una (4)connessione possono essere attivate o per chiudere una (4)connessione al termine del trasferimento delle PDU o per rifiutare l'apertura di una (4)connessione con una QOS che non si è in grado di garantire. Questo rifiuto può arrivare sia dal fornitore del servizio sia dalla (5)entità chiamata.

È possibile descrivere con un diagramma a stati la successione delle primitive ad un certo (4)SAP. Il diagramma è riportato nella figura 9.2; lo stato 1 è lo stato libero, lo stato 2 è lo stato di connessione uscente in corso, il 3 lo stato di connessione entrante in corso e lo stato 4 è il trasferimento dati.

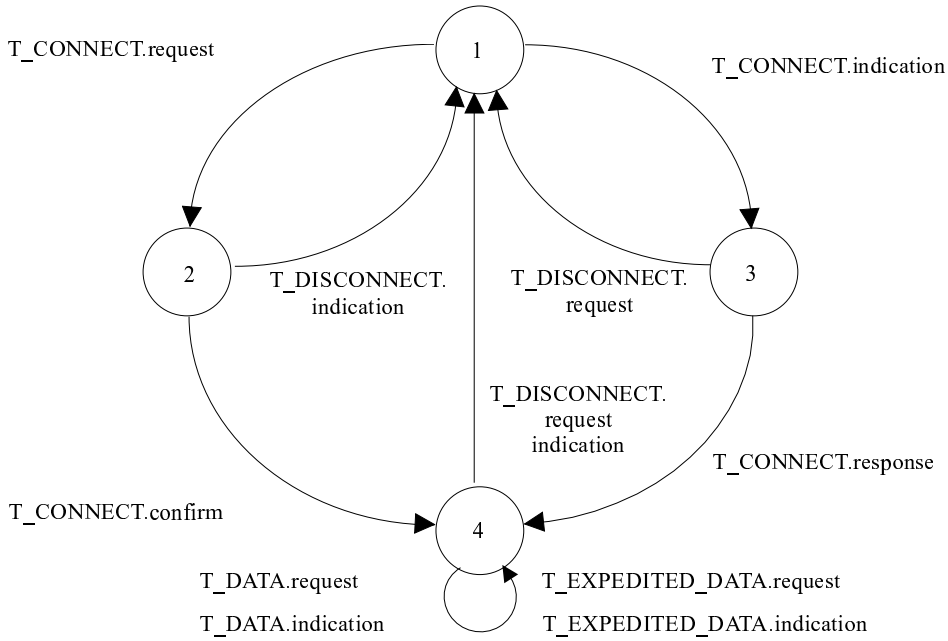


Figura 9.2. Diagramma a stati

9.3 Controllo di flusso

Il controllo di flusso del livello trasporto sfrutta un protocollo a finestra con una dimensione della finestra di trasmissione WT variabile. La dimensione della WT è determinata dal ricevitore, che abilita il trasmettitore mediante un parametro detto *credito* all'interno delle (4)PDU di ACK. La dimensione della finestra quindi dipende dall'ultimo ACK ricevuto.

Nei protocolli a finestra normali le due finestre di trasmissione e ricezione hanno dimensione fissa e si spostano sulla base delle PDU di dati o ACK ricevute. In questo caso invece, il ricevitore invia con gli ACK anche un nuovo valore per la dimensione della finestra del trasmettitore; il nuovo valore potrà essere o maggiore del precedente o minore e in questo caso non potrà che decrescere di una unità (se le conferme cumulative non sono ammesse). Infatti si può tenere fermo il limite superiore della finestra del trasmettitore ma non farlo diminuire.

La finestra del trasmettitore quindi oltre a spostarsi può variare dimensione sulla base del parametro credito degli ACK. Se il credito fosse 0, il trasmettitore sarebbe bloccato. In questo modo il ricevitore riesce a effettuare il controllo di flusso.

Un esempio di funzionamento è illustrato nella figura 9.3. Nella parte sinistra è rappresentata una situazione in cui le PDU sono numerate su 3 bit, la finestra del trasmettitore è posizionata sugli intervalli 0, 1 e 2, mentre la finestra del ricevitore è posizionata sull'intervallo 0. Quando al ricevitore arriva la PDU 0, esso fa avanzare di una unità la WR e manda un ACK con un parametro di credito. Supponendo che tale credito valga 2, quando l'ACK viene ricevuto dal trasmettitore, esso, come si vede nella parte destra della figura, sposta in avanti di una unità il limite inferiore della propria finestra di trasmissione, senza modificare il limite superiore. Non sarebbe stato possibile invece inviare un credito minore di 2, perché in questo caso il trasmettitore avrebbe dovuto diminuire il limite superiore della WT.

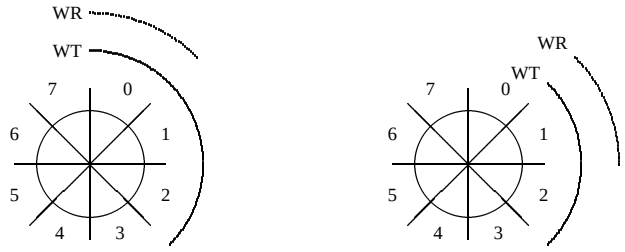


Figura 9.3. Esempio di funzionamento del parametro credito

9.4 Formati delle (4)PDU

Il formato di una (4)PDU è riportato nella figura 9.4.

LI	PARTE FISSA	PARTE VARIABILE	DATI

Figura 9.4. Formato di una (4)PDU

Il campo LI è l'indicatore della lunghezza dell'intestazione della (4)PDU (compreso il byte LI stesso).

La parte fissa dell'intestazione dipende dal tipo di (4)PDU.

La parte variabile dell'intestazione è caratterizzata da una terna di campi per ogni parametro indicato: il primo campo descrive il codice del parametro, il secondo la lunghezza e il terzo il valore assunto dal parametro. In questo modo l'ordine dei parametri che compongono la parte variabile (per esempio quelli che specificano la QOS) non è fisso, ma definibile dall'utente. Nella parte variabile, nel caso di classe di trasporto 4 si possono inserire 16 bit di un codice ciclico per il controllo degli errori.

I tipi di (4)PDU più importanti sono:

- CR, Connection Request;

- CC, Connection Confirm;
- DR, Disconnection Request;
- DC, Disconnection Confirm;
- DT, DaTa;
- ED, Expedited Data;
- AK, AcKnowledge;
- EA, Expedited Acknowledge;
- ER, ERror;
- RJ, ReJect;

Esaminiamo il formato di alcune (4)PDU.

Connection Request

Il formato della (4)PDU di tipo Connection Request è riportato nella figura 9.5.

Figura 9.5. (4)PDU Connection Request

Il codice 1110 identifica il tipo di CR, il campo CDT è il valore del credito utilizzato nelle classi che utilizzano il controllo di errore e di flusso come descritto precedentemente. Il DESTINATION REFERENCE è a zero perché il chiamante identifica solo il numero della connessione nel SOURCE REFERENCE. Il campo CLASSE identifica la classe del protocollo di trasporto. Il campo OPZIONI specifica se il controllo di flusso è utilizzato e se le (4)PDU hannno un formato normale o esteso come numerazione.

La parte variabile può contenere molte informazioni:

- un source e un destination address, ovvero gli indirizzi dei (4)SAP;
- la dimensione della (4)PDU (minima 128 byte, massima 8192);
- checksum per il controllo di errore se il protocollo è di classe 4;
- tutti i parametri che definiscono la qualità del servizio;
- le classi alternative utilizzabili se non si riesce ad utilizzare la classe prescelta.

La parte dati è opzionale e al massimo può essere lunga 32 byte.

Connection Confirm

La (4)PDU di tipo Connection Confirm è identica a quella di tipo Connection Request. In questo caso però diventa importante il campo DESTINATION REFERENCE.

Dal momento in cui la (4)entità chiamata riceve la (4)PDU di tipo Connection Confirm, per la trasmissione delle unità dati si utilizzeranno solo gli indicatori di connessione. Gli indirizzi dei (4)SAP non sono più necessari.

Data

Esaminiamo il formato della (4)PDU per la trasmissione dei dati per le classi 2, 3 e 4, riportato nella figura 9.6.

LI	11110000	DEST REF	EOT	TPDU NUM	VARIABILE	DATI

Figura 9.6. (4)PDU dati

Il campo DESTINATION REFERENCE contiene l'identificatore della connessione. Il campo TPDU NUM contiene il numero d'ordine della (4)PDU e il bit EOT indica se la PDU è l'ultima della sequenza. La parte variabile può contenere un checksum per il controllo degli errori. La parte dati è limitata dalla dimensione massima della (4)PDU stabilita nella fase di apertura di una connessione.

Il formato esteso prevede l'uso di quattro byte per il campo TPDU NUM, consentendo l'uso di 31 bit per la numerazione delle (4)PDU; il formato normale utilizza un solo byte.

Acknowledge

Il formato della (4)PDU di ACK per le classi 2, 3 e 4 è riportato nella figura 9.7. A differenza di quanto si verificava nel livello 2, le conferme devono essere inviate in modo esplicito; non esiste la possibilità di inviarle in modo implicito sfruttando le (4)PDU dati.

LI	0110 CDT	DEST REF	TPDU ATTESA	VARIABILE	DATI

Figura 9.7. (4)PDU di ACK

Expedited Data e Acknowledge

I dati e le conferme veloci sono smaltiti con priorità maggiore rispetto ai dati normali e devono essere utilizzati solo per trasferire messaggi urgenti; un esempio tipico sono le informazioni di controllo nel caso di situazioni di congestione della rete; il protocollo è di tipo *stop-and-wait.*

I formati sono sostanzialmente uguali a quelli delle (4)PDU dati e ACK normali. La numerazione delle (4)PDU di tipo expedited è separata rispetto a quella delle PDU e degli ACK normali. Il campo dati al massimo può essere lungo 16 byte.

10

La rete Internet

10.1 Introduzione

10.1.1 Cos'è Internet?

La rete Internet è una rete di telecomunicazioni a pacchetto, con topologia a maglia irregolare, che permette l'interconnessione a livello mondiale di un numero sempre crescente di utenti. La rete Internet è fisicamente composta da host e router, collegati mediante reti eterogenee del tutto indipendenti: LAN, MAN, canali punto-punto in fibra ottica o in cavo coassiale, ponti radio, rete ISDN, reti Frame Relay, reti ATM. Per questo motivo si dice che Internet è una rete che interconnette sottoreti. Vengono dati alcuni elementi di nomenclatura che serviranno in seguito:

- **host (o terminale):** un calcolatore collegato alla rete Internet cui abbiano accesso un certo numero di utenti, sul quale vengono eseguiti processi applicativi;
- **router:** è un nodo di commutazione ed instradamento, cioè un nodo intermedio con la capacità di decidere verso quale prossimo dispositivo instradare un pacchetto;
- **client e server:** ogni host può comportarsi come un client, ovvero come un calcolatore in cerca di servizi forniti dalla rete Internet, oppure da server, un calcolatore che fornisce servizi agli utenti della rete Internet;
- **sottorete:** un insieme di host e zero o più router caratterizzato dal fatto che gli host della sottorete si scambiano i pacchetti direttamente (senza passare dal router, che fornisce invece connettività verso altre sottoreti);
- **autonomous system:** un insieme di host e router, che appartengono a una o più sottoreti, sotto la stessa autorità amministrativa.

10.1.2 La storia di Internet

Nella prima metà degli anni '70 la Defence Advanced Research Project Agency (DARPA) dimostrò interesse per lo sviluppo di una rete a commutazione di pacchetto per l'interconnessione di calcolatori eterogenei, da utilizzarsi come mezzo di comunicazione tra università, laboratori di ricerca e centri governativi e del

Ministero della Difesa degli Stati Uniti. DARPA finanziò a tal scopo l'Università di Stanford e la BBN (Bolt, Beranek and Newman) affinché sviluppassero un insieme di protocolli di comunicazione.

Verso la fine degli anni '70, tale sforzo portò al completamento dell'*Internet Protocol Suite*, i cui protocolli più noti sono il TCP (*Transmission Control Protocol*) e l'IP (*Internet Protocol*).

Questi protocolli furono utilizzati da un gruppo di ricercatori per la rete ARPAnet e ottennero un elevato successo, sia perché posti immediatamente nel dominio pubblico e quindi utilizzabili gratuitamente da tutti, sia perché sono stati subito adottati in modo nativo dal sistema operativo Unix.

Gli anni '90 vedono l'esplosione della popolarità di Internet al di fuori dell'ambito accademico, di ricerca e soprattutto anche al di fuori degli Stati Uniti. Con il suo tasso di crescita del 5% al mese, oggi l' architettura di rete è predominante ed ha oscurato il modello ISO/OSI proprio negli anni che dovevano segnare la sua maturità.

10.2 Architettura di protocolli

Il nome più accurato per l'architettura di rete rimane quello di Internet Protocol Suite, anche se comunemente si fa riferimento ad essa con la sigla TCP/IP o IP/TCP. Questo può portare ad alcune ambiguità (si veda la figura 10.1): ad esempio è comune sentir parlare di NFS come un servizio basato su TCP/IP, anche se NFS non usa il protocollo TCP, ma un protocollo alternativo detto UDP appartenente all'Internet Protocol Suite. Visto l'uso estremamente comune della sigla TCP/IP, essa verrà adottata anche in queste dispense, quando non si rischi di creare confusione.

I protocolli appartenenti a questa architettura sono soggetti ad un processo di evoluzione coordinato dall'IETF (Internet Engineering Task Force); tutte le proposte e le eventuali modifiche apportate a tali protocolli sono raccolte in documenti chiamati RFC (*Request for Comments*) facilmente reperibili sulla rete Internet. Famoso è lo RFC 791 Internet Protocol, datato 1981, che specifica appunto il protocollo IP.

La figura 10.2 mostra l'architettura dell'Internet Protocol Suite e la paragona con il modello di riferimento ISO/OSI; è possibile anche leggere le diverse pile alternative di protocolli utilizzate per le diverse applicazioni.

Nella figura 10.3 sono presentati gli strati principali di questa architettura, assieme alla nomenclatura normalmente utilizzata per le unità dati:

- **Application:** si interfaccia direttamente con il livello transport, scegliendo il protocollo più opportuno (UDP o TCP) a seconda della applicazione. Definisce i protocolli necessari alla gestione dei servizi offerti agli utenti.

- **Transport:** fornisce la comunicazione end-to-end, ovvero tra utente e utente, regolando la velocità di trasporto dell'informazione. Se il protocollo utilizzato è il protocollo TCP, garantisce anche un trasporto affidabile (controllo di errore, di flusso e di sequenza).

- **Internet:** si occupa dell'instradamento (routing). Quando riceve un datagram, ne controlla la validità (ma solo dell'intestazione, non del contenuto) e decide se riceverlo o instradarlo verso un altro router.

- **Network interface:** comprende una famiglia di protocolli specifici a seconda della rete che si considera e volti al trasferimento di unità dati all'interno di tale rete.

Nella figura 10.1 sono presentate diverse pile possibili, quali ad esempio telnet/TCP/IP/Ethernet, FTP/TCP/IP/Ethernet, SNMP/UDP/IP/FDDI o NFS/XDR/RPC/UDP/IP/Token-Ring.

In breve le caratteristiche principali dell'architettura possono essere schematizzate nei seguenti tre punti:

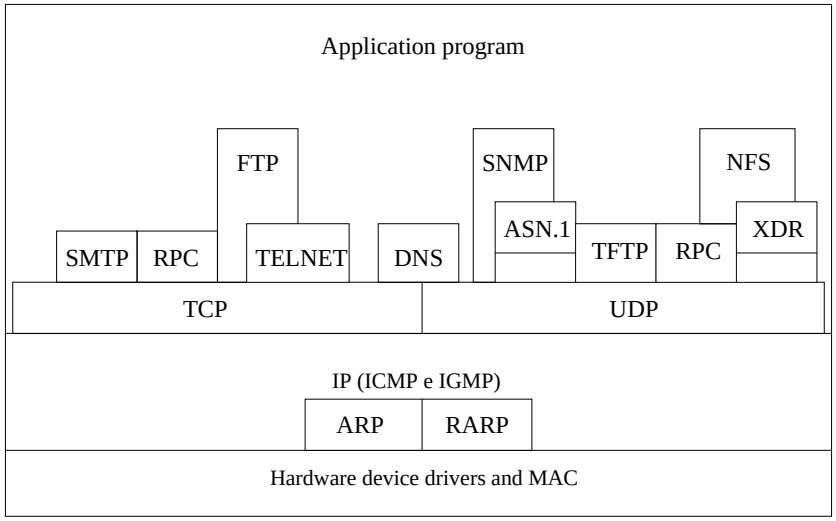


Figura 10.1. Architettura TCP/IP

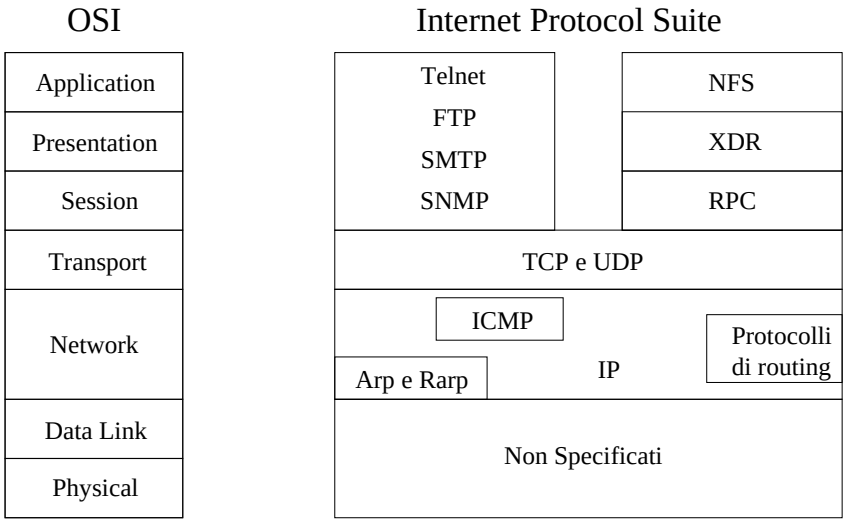


Figura 10.2. Confronto ISO/OSI con TCP/IP

- il controllo di errore è effettuato opzionalmente end-to-end a livello 4 (il livello transport) ma non all'interno della rete, per aumentare la velocità riducendo i tempi di elaborazione ai nodi di commutazione intermedi;
- a livello rete, detto livello Internet, la comunicazione è di tipo *connectionless* non è quindi affidabile: i

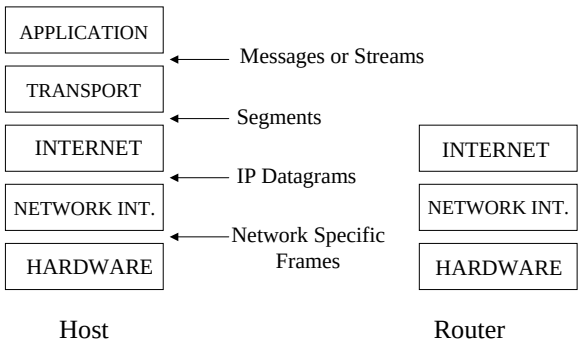


Figura 10.3. Strati principali della architettura TCP/IP

pacchetti corrotti e quelli che non possono essere trasmessi o ricevuti per congestione sono scartati;

- gli host non sono semplici, come nel caso della rete Itapac basata su X.25. Devono partecipare al funzionamento della rete: “ascoltano” i segnali di feedback, reagiscono a messaggi di routing e cooperano con i router per ottenere un buon funzionamento della rete.

10.3 Il livello 4: UDP Protocol (RFC 768)

Lo User Datagram Protocol (UDP) è un protocollo di trasporto, alternativo a TCP, di tipo connectionless e unreliable. UDP è un protocollo molto più semplice di TCP ed è utilizzato quando l’affidabilità di TCP non è richiesta. Non utilizza ACK, non ordina messaggi, non fornisce controllo di flusso.

Un applicativo che utilizzi il protocollo UDP deve preoccuparsi di controllare l’affidabilità della connessione e risolvere eventualmente i seguenti problemi:

- perdita o duplicazione di messaggi;
- ricezione di messaggi non in ordine;
- ricezione di messaggi a velocità troppo elevata rispetto alla capacità del ricevitore.

10.3.1 (De)multiplazione e meccanismo delle porte di UDP

La funzione principale di UDP è detta “multiplexing / demultiplexing”.

UDP utilizza IP per trasportare i messaggi tra host. Rispetto al protocollo IP, aggiunge la capacità di distinguere tra destinazioni (programmi applicativi) diverse sullo stesso host. In trasmissione, accetta i messaggi provenienti dai vari applicativi e li passa al livello IP (funzione di multiplexing). In ricezione, raccoglie i datagram provenienti dal livello IP e li inoltra all’applicativo opportuno (funzione di demultiplexing).

Poiché il destinatario di un pacchetto non è un host ma uno dei processi attivati dal sistema operativo, è ambiguo identificare la destinazione con uno di tali processi per varie ragioni:

- i processi sono creati e distrutti dinamicamente;

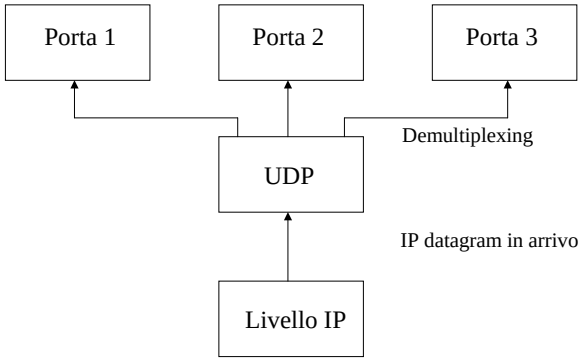


Figura 10.4. Il concetto di protocols-port in UDP

- devono poter essere sostituiti senza informare tutti gli interessati alla comunicazione (il reset di un host può cambiare tutti i processi);
- chi trasmette è interessato alla funzione realizzata dal processo più che al processo in sé (ad esempio un utente remoto deve poter contattare un file server senza conoscere quale processo implementa tale funzione).

Per risolvere questo problema ciascun host contiene un insieme di punti di destinazione astratti detti **protocol ports**, identificati da un numero intero. A tali protocol ports corrispondono servizi più che processi. Nella figura 10.4 è schematizzato un esempio di demultiplexing effettuato da UDP utilizzando il concetto di protocol ports.

Esistono due approcci per l'assegnazione dei numeri alle relative porte:

- **universal assignment:** vengono definiti ufficialmente e riconosciuti da tutti;
- **dynamic binding:** ogni volta che un programma ha bisogno di una porta, questa viene assegnata dal sistema operativo. Per conoscere da un host remoto l'assegnazione corrente è necessario richiedere su quale porta il servizio desiderato è disponibile.

I valori più piccoli sono definiti universalmente, come illustrato nella tabella seguente. Altre porte sono assegnate al protocollo TCP: si veda il paragrafo 1.4.10.

UDP PORT	SERVIZI
7	echo
13	daytime
17	quote (citazione del giorno)
37	time
11	systat (utenti attivi)
15	netsat
19	chargen
42	name (host name)

10.3.2 Formato dello user datagram di UDP

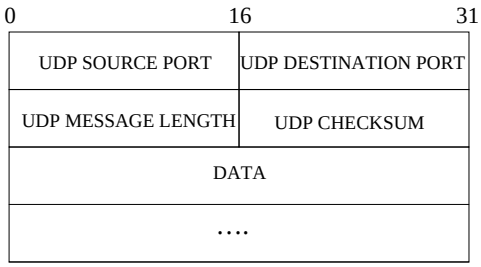


Figura 10.5. Formato dello user datagram utilizzato da UDP

L'unità di trasferimento è chiamata user datagram (si veda la figura 10.5). L'intestazione è divisa in 4 campi di 16 bit che specificano:

- **source port:** campo opzionale che indica la porta dalla quale il messaggio è stato spedito (posto a zero se non utilizzato);
- **destination port:** la porta a cui il datgram è destinato;
- **length:** la lunghezza del datagram;
- **checksum:** campo utilizzato opzionalmente per effettuare un controllo di errore su tutto il pacchetto (posto a zero se non utilizzato).

Il campo length specifica il numero di ottetti dell'UDP datagram, includendo sia l'intestazione che i dati; il minimo valore che può assumere è quindi 8, cioè la lunghezza della sola intestazione.

10.3.3 Il campo checksum e lo pseudo-header di UDP

Il calcolo del checksum avviene su tutti i byte del pacchetto UDP più un certo numero di byte che non appartengono al pacchetto e che vengono detti pseudo-header. Lo pseudo-header contiene gli indirizzi IP di destinazione e sorgente, ma non viene trasmesso con l'UDP datagram e non è conteggiato nel campo length. L'uso dello pseudo-header permette di verificare che l'UDP datagram abbia raggiunto la corretta destinazione, determinata dall'host, ovvero dal suo indirizzo IP, e dalla porta di destinazione.

Lo pseudo header utilizzato nel calcolo del checksum, è formato dai seguenti campi (si veda la figura 10.6):

- **source IP address e destination IP address**;
- **proto:** tipo di protocollo utilizzato (17 per UDP);
- **UDP length:** lunghezza dell'UDP datagram (escludendo lo pseudo header).

Per verificare il checksum, il ricevitore deve estrarre questi campi dall'IP header, crearsi lo pseudo header, allinearli su 16 bit ed effettuare il complemento ad uno della somma. Ciò viola il principio di **indipendenza** tra i livelli: non si seguono i concetti di stratificazione.

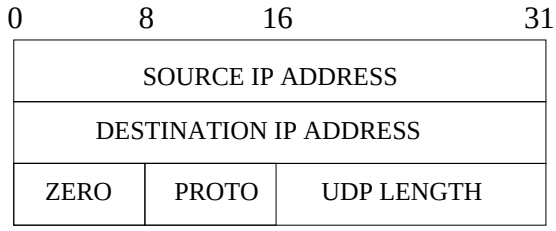


Figura 10.6. Formato dello pseudo-header di UDP

Si deve inoltre precisare che esiste una possibile ambiguità sul valore assunto dal campo checksum, poiché non è escluso che il risultato del calcolo del checksum sia zero. Si ricordi infatti che, nel caso in cui il controllo d'errore non sia utilizzato, il campo checksum deve essere posto a zero. In realtà l'aritmetica in complemento a uno ha due rappresentazioni dello zero: tutti i bit a 0 o tutti a 1. Quando il checksum calcolato è zero, si sceglie la seconda.

10.4 Il livello 4: TCP Protocol (RFC 793)

Il TCP è un protocollo di trasporto, come UDP, però di tipo orientato alla connessione, e fornisce un servizio di tipo full-duplex (bidirezionale - contemporaneo), con acknowledge (conferma) e controllo di flusso.

TCP è un protocollo adattabile anche a reti che non utilizzano il protocollo IP a livello rete: non fa alcuna ipotesi sull'affidabilità dei livelli inferiori, supponendo di avere a disposizione un semplice servizio di tipo datagram con possibilità di:

- perdita dei pacchetti (per errori di trasmissione, code piene, guasti hardware);
- arrivi fuori sequenza;
- ritardi variabili;
- duplicazione dei pacchetti.

L'obiettivo del protocollo TCP è fornire un servizio affidabile e orientato alla connessione agli applicativi di livello superiore per sollevarli dalle problematiche di gestione della comunicazione di rete. Il servizio di trasferimento offerto da TCP è caratterizzato essenzialmente da:

- **stream orientation:** i dati sono visti come stream di bit organizzati in ottetti; il processo TCP al ricevitore restituisce all'applicativo esattamente lo stesso stream di ottetti passati dall'utente del nodo sorgente al processo TCP di trasmissione.
- **virtual circuit connection**: prima di trasferire i dati i due processi devono stabilire una **connessione**. Chi inoltra la richiesta di apertura deve aspettare la conferma per inviare i dati; è previsto un meccanismo di controllo per la corretta ricezione e per rivelare la caduta della connessione. Il termine circuito virtuale sottolinea che i processi vedono la connessione come un circuito dedicato, ma la rete non riserva risorse dedicate alla connessione;

- **trasferimento bufferizzato e stream non strutturato:** sugli host i processi generano e ricevono messaggi, di dimensioni dipendenti dalle applicazioni; questi vengono inseriti in buffer (code) e prelevati da TCP che può, per motivi di efficienza, riorganizzare la trasmissione in segmenti di dimensione opportuna. Ad esempio, se l'applicazione genera dati di piccole dimensioni, TCP può aspettare finché può trasmettere un segmento di dimensione ragionevole. Le applicazioni hanno la possibilità di forzare il trasferimento di tutti i dati generati senza aspettare il riempimento del buffer e consentire a questi di giungere al ricevitore senza ritardi per applicazioni interattive. Non è possibile per una applicazione conoscere la dimensione dei segmenti scelta da TCP poiché questa è variabile. La suddivisione in segmenti non corrisponde ad una strutturazione del flusso di dati: è necessario che gli applicativi si accordino su un formato per lo stream affinché il suo contenuto sia intelligibile.
- **connessione full duplex:** TCP consente il trasferimento bidirezionale dei dati (full duplex): esistono sempre due flussi indipendenti di informazione in direzioni opposte, senza apparente interazione. Questo tipo di connessione consente di inviare le informazioni di controllo per il flusso in un senso utilizzando i pacchetti che viaggiano in senso inverso: il meccanismo è detto **piggybacking** e riduce il traffico di rete evitando il trasferimento di pacchetti specifici per il controllo.

10.4.1 Porte, connessioni e endpoints di TCP

TCP consente la comunicazione contemporanea di più applicativi trasferendo i messaggi che raggiungono l'host al processo cui sono destinati. La destinazione finale è identificata sempre da un **protocol port number**, in analogia a quanto visto per UDP nel paragrafo 1.3.1. Ciascun processo è individuato da un endpoint, una coppia di numeri interi (host_address, port_number), dove host_address rappresenta l'indirizzo IP dell'host e port_number è il TCP port number; questa coppia spesso viene chiamata *socket.* TCP definisce sempre una connessione, che sarà individuata quindi da una coppia di endpoint.

Ad esempio, una connessione attiva tra l'host con indirizzo IP (128.9.0.32) all'Information Science Institute e l'host (128.10.2.3) presso la Purdue University, è definita, per esempio, da:

$$(128.9.0.32,1184) - (128.10.2.3,53)$$

Contemporaneamente potrebbe essere attiva la connessione:

$$(128.2.254.139,1184) - (128.10.2.3,53)$$

che ha in comune con la prima la porta 53 sull'host (128.10.2.3): questo è possibile poiché per TCP si tratta di due connessioni diverse. Questo consente ai programmi di fornire lo stesso servizio sullo stesso port number di uno stesso host ad una molteplicità di connessioni.

Assegnazione delle porte in TCP

Come è stato già citato per UDP, esistono due approcci fondamentali per l'assegnazione dei numeri relativi alle porte:

- **universal assignement:** vengono definiti ufficialmente e riconosciuti da tutti;
- **dynamic binding:** ogni volta che un programma ha bisogno di una porta, questa viene assegnata dal sistema operativo o dal software di rete.

TCP combina l'assegnazione statica universale dei port number con il dynamic binding. Per i processi comunemente invocati si usano numeri di porta noti, lasciando un insieme di porte disponibili al sistema operativo per fornirle agli altri programmi:

7	echo
11	systat
13	daytime
17	quote
19	chargen
21	ftp
23	telnet
25	smtp
37	time
79	finger
80	http

10.4.2 Meccanismo a finestra in TCP

TCP utilizza un protocollo a finestra per trasmettere i segmenti in modo affidabile e impedire:

- perdite di dati;
- errori sui dati (controllo CRC su ogni segmento);
- duplicazioni;
- arrivi fuori sequenza.

Il trasmettitore invia segmenti numerandoli in modo sequenziale per poterli distinguere. La numerazione in TCP è a livello di ottetti e non di segmenti: ogni segmento contiene il numero di sequenza del primo byte contenuto nel segmento. Il ricevitore invia conferme (ACK) a fronte del ricevimento di un segmento; anche gli ACK sono numerati sequenzialmente. Gli ACK specificano il numero di sequenza del primo byte mancante al ricevitore, cioè quello che si è in attesa di ricevere. Gli ACK inoltre sono **cumulativi**, cioè confermano la ricezione di tutti gli ottetti fino a quello indicato.

Una volta trasmesso il segmento, il trasmettitore fa partire un orologio che determina il tempo massimo di attesa della conferma del pacchetto; i ritardi della rete possono far scadere l'orologio (timeout) prima che sia ricevuta la conferma. In tal caso il segmento deve essere ritrasmesso. Per migliorare lo sfruttamento della banda disponibile TCP usa un meccanismo a finestra (**sliding window**), che consente la trasmissione di più pacchetti prima di ricevere le conferme (tutti quelli compresi nella dimensione della finestra). Per conservare tutte le informazioni che gli servono, al trasmettitore sono definiti tre puntatori che segnano rispettivamente il limite sinistro della finestra, il più recente ottetto trasferito e il limite destro della finestra (l'ottetto con il più alto numero di sequenza che può essere trasferito senza ricevere la prima conferma, cioè la dimensione della finestra).

Quando viene confermato il segmento più vecchio, la finestra trasla a destra verso numeri di sequenza più alti, consentendo l'invio di un ulteriore segmento. Il trasmettitore si comporta come nel protocollo Go-Back-N.

Anche il ricevitore dispone di una finestra di dimensione maggiore di uno per ricostituire correttamente il flusso dati, anche nel caso in cui i segmenti arrivino fuori sequenza (si veda la figura 10.7).

10.4.3 Controllo di flusso e di congestione in TCP

I controlli di flusso e di congestione effettuati da TCP vengono realizzati dimensionando opportunamente la finestra di trasmissione sopra definita. Regolando dinamicamente la dimensione della finestra durante una connessione infatti si regola la velocità di trasmissione del protocollo TCP.

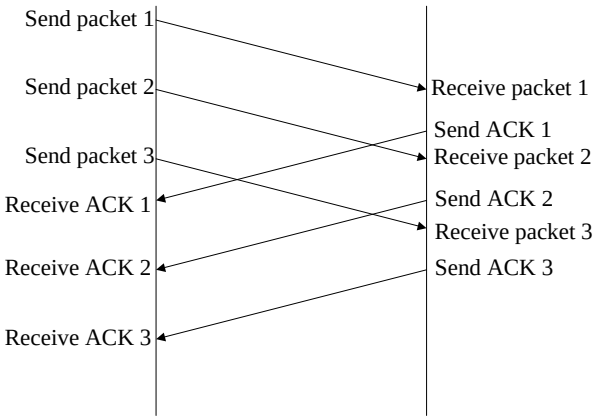


Figura 10.7. Meccanismo a finestra in TCP

Il controllo di flusso permette di evitare la trasmissione di dati ad una velocità superiore a quella per cui il ricevitore è in grado di ricevere.

Il controllo di congestione permette di evitare di congestionare la rete. Una rete si congestiona nei momenti in cui il traffico in ingresso ad un router supera la capacità di trasmissione su uno dei canali in uscita. L'effetto della congestione è quello di aumentare il numero di pacchetti in attesa di trasmissione, arrivando alla perdita di pacchetti per la dimensione finita dei buffer presenti nei router. La perdita di pacchetti a sua volta comporta ritrasmissioni, ovvero un aumento del carico di rete.

Il controllo di flusso in TCP è ottenuto con una variazione della dimensione della finestra di trasmissione indotta dal ricevitore. Gli ACK contengono un campo chiamato *window advertisement* che indica la nuova dimensione della finestra disponibile in ricezione, ovvero lo spazio di memoria libero (non occupato da segmenti ricevuti ma non ancora "passati" all'applicativo) che il ricevitore ha dedicato alla connessione. Se il ricevitore è lento, lo spazio di memoria libero si riduce e di conseguenza la finestra disponibile in ricezione viene ridotta. La finestra del trasmettitore non può mai superare la dimensione della finestra disponibile in ricezione. Se questa viene ridotta, sarà ridotta anche la finestra del trasmettitore.

La congestione è un fenomeno che tende ad autoalimentarsi: la crescita dei ritardi crea perdita di pacchetti e i protocolli che usano meccanismi di controllo a finestra come TCP rispondono alle perdite con la ritrasmissione dei pacchetti, incrementando ulteriormente il traffico. In TCP la determinazione della situazione di congestione avviene per decisione autonoma da parte del trasmettitore, a fronte della perdita di un pacchetto. Per contrastare i fenomeni di congestione, TCP riduce la velocità di trasmissione non appena si rende conto che si è verificata una perdita di pacchetti. Per controllare la congestione nella rete Internet, nella versione TCP si definiscono una *congestion window* (*cwnd*) ed una *soglia*. La *cwnd*, finestra corrente di trasmissione, avrà sempre una dimensione inferiore alla massima dimensione della finestra del ricevitore ($w_{\max}$), per soddisfare il controllo di flusso. La $w_{\max}$ è negoziata nella fase di instaurazione della connessione.

Inizialmente la *cwnd* ha una dimensione pari ad 1 segmento di dimensione massima per la connessione e la *soglia* è pari a $w_{\max}$. Se non si crea congestione, la *cwnd* cresce inizialmente secondo una tecnica detta **slow-start**: per ogni ACK ricevuto la *cwnd* è incrementata di un segmento. Ciò corrisponde ad una crescita esponenziale della *cwnd*.

Quando la *cwnd* raggiunge una dimensione pari alla *soglia*, la crescita rallenta e si entra nella fase di

congestion avoidance; l'incremento di un segmento della *cwnd* avviene solo quando tutti i segmenti della finestra sono stati confermati. Si ha crescita lineare della *cwnd.*

Se non si hanno perdite, raggiunto il valore $w_{\max}$, la *cwnd* non cresce ulteriormente.

Il trasmettitore TCP suppone che la rete sia congestionata nel momento in cui si verifica una perdita di segmento. La perdita di un segmento è rilevata nei due seguenti casi:

- scadenza di un timeout al trasmettitore;
- ricezione di 4 ACK uguali consecutivi (3 ACK duplicati).

Quando ciò si verifica, il trasmettitore reagisce:

- portando la *soglia* a metà della *cwnd* corrente e la *cwnd* nuova ad 1 segmento (fase di slow start).
- portando la *soglia* e la *cwnd* ad un valore pari alla metà della finestra corrente (fase di congestion avoidance).

10.4.4 Determinazione del timeout

TCP deve determinare un valore ragionevole da assegnare al timeout per funzionare in modo efficiente. È necessario a tale scopo la conoscenza di una stima del round trip time. Purtroppo il round trip time è fortemente variabile nella rete Internet a causa della variabilità del traffico e della diversità di percorsi che i vari segmenti seguono all'interno della rete.

Per stimare il valore del round trip time, per ogni segmento viene calcolata la differenza $last\_RTT\_sample$ di tempo tra l'istante di invio e di ricezione dell'ACK corrispondente; questo valore, opportunamente pesato con un coefficiente α aggiorna la stima del round trip time (RTT). Il timeout è poi posto ad un valore maggiore (normalmente doppio) di tale stima:

$$\begin{aligned} RTT &= \alpha \times RTT + (1-\alpha) \times last\_RTT\_sample \\ TIMEOUT &= \beta \times RTT \\ \beta &> 1 \end{aligned}$$

Possono nascere problemi nel caso di ACK cumulativi e nel caso di ritrasmissioni. Nel caso di ACK cumulativi, questi non sono associabili in modo specifico ad un segmento. Inoltre, nel caso in cui si utilizzino per la misura ACK relativi a segmenti ritrasmessi, è necessario decidere se associare la ricezione dell'ACK all'invio della prima o dell'ultima copia del segmento ritrasmesso. Se si associa l'ACK al primo segmento inviato, il round trip time stimato RTT potrebbe crescere a dismisura nelle situazioni di perdita continua per congestione. Se si associa invece l'ACK all'ultimo segmento trasmesso nascono problemi quando si verifica un improvviso aumento dei ritardi tra due trasmissioni consecutive dello stesso segmento. Se si riceve l'ACK relativo alla prima trasmissione in ritardo, a causa dell'aumento dei round trip time, ma lo si associa alla seconda trasmissione, si stima un round trip time piccolo e si determina di conseguenza un timeout insufficiente rispetto alla mutata situazione di rete. Tale stima crea un nuovo duplice invio del segmento seguente, con analogo effetto; tale situazione rischia di ripetersi nel tempo.

Determinazione del timeout: algoritmo di Karn

Per risolvere i problemi del calcolo del round trip time TCP utilizza il *Karn's algorithm*: i campioni relativi a segmenti che sono stati ritrasmessi non aggiornano la stima. Tale regola da sola sarebbe insufficiente: infatti, se si verifica un aumento del ritardo, il segmento deve essere ritrasmesso poiché scade il timeout; gli ACK ricevuti non sono utilizzabili per aggiornare la stima del round trip time che rimane inferiore al round trip time reale. Per evitare questo problema si utilizza una strategia di backoff esponenziale sul valore assegnato al timeout dopo ogni ritrasmissione:

$$TIMEOUT = \gamma \times TIMEOUT$$

dove γ assume un valore pari a 2.

La determinazione del valore da assegnare al timeout è separata dalla stima del round trip time. Se si deve ritrasmettere si aumenta il valore del timeout per ogni trasmissione senza ricezione di ACK, finché il segmento non viene trasferito con successo; la stima del round trip time non viene modificata. Non appena la trasmissione di un segmento ha successo si utilizza la nuova misura del round trip time per aggiornare la stima del round trip time.

Nelle versioni più recenti del protocollo, per aggiornare la stima del round trip time si utilizza una formula più complessa che tiene conto anche della varianza delle misure:

$$\begin{aligned} DIFF &= last\_RTT\_sample - RTT \\ Smoothed\_RTT &= RTT + \delta \times DIFF \\ DEV &= DEV + \rho\,(|DIFF| - DEV) \\ TIMEOUT &= Smoothed\_RTT + \eta \times DEV \end{aligned}$$

dove DEV è la deviazione media stimata, δ è un coefficiente compreso tra 0 e 1 che controlla quanto velocemente il nuovo campione ha effetto sulla media pesata, ρ è un coefficiente tra 0 e 1 che controlla quanto velocemente il nuovo campione ha effetto sulla deviazione media e η è un fattore che controlla quanto la deviazione ha effetto sul timeout.

"Silly window syndrome" in TCP

È un problema che porta alla trasmissione di segmenti piccoli con uno sfavorevole rapporto intestazione/dati se il ricevitore è lento e legge i dati a piccoli blocchi (ad esempio un ottetto alla volta). Se il processo in trasmissione genera dati velocemente il buffer del ricevitore tende a riempirsi specificando negli ACK una dimensione della finestra sempre più piccola, al limite nulla, per interrompere momentaneamente il flusso di dati. A questo punto nel buffer si liberano spazi di un singolo byte (ogni volta che l'applicazione effettua una lettura) che forzano la trasmissione di un segmento contenente solo un ottetto di dati (quindi con un rapporto header-dati elevato). Lo stesso fenomeno si verifica quando il processo in trasmissione genera dati in blocchi piccoli: se la rete è veloce, TCP tende a trasmettere il contenuto del buffer non appena trova dati nel buffer si occupa nuovamente il canale in modo inefficiente con segmenti piccoli.

Le implementazioni recenti del protocollo TCP adottano tecniche per prevenire questo comportamento. Il trasmettitore ritarda l'invio del segmento finché non ha accumulato una quantità ragionevole di dati (tecnica detta *clumping*); il ricevitore comunica la dimensione aggiornata della propria finestra solo quando questa ha raggiunto dimensioni opportune.

“Receive-side silly window” in TCP

Sono possibili due approcci per risolvere il problema al ricevitore; il ricevitore TCP può:

- confermare ciascun segmento che arriva a destinazione senza aggiornare la finestra disponibile fino a quando questa sia pari ad almeno metà della memoria disponibile in ricezione, oppure sia pari alla dimensione massima di un segmento (MSS);
- ritardare l’invio della conferma che sarà cumulativa per tutti i segmenti ricevuti.

I vantaggi della seconda tecnica sono:

- vengono spediti meno ACK, quindi si riduce il traffico;
- se l’applicazione genera una risposta non appena i dati arrivano, il ritardo consente il piggybacking mandando la conferma insieme al segmento trasmesso;
- nei casi in cui il ricevitore legge i dati non appena arrivano, un piccolo ritardo permette l’invio di un segmento che conferma il dato e comunica già la nuova dimensione della finestra.

Lo svantaggio principale è dato dal rischio di ritrasmissione se la conferma viene ritardata troppo; inoltre, l’istante di arrivo degli ACK è usato per la stima del RTT, che viene alterata. Per questi motivi esiste un limite massimo per il ritardo pari a 200 ms.

“Send-side silly window avoidance (Nagle Algorithm)” in TCP

La tecnica di clumping consiste nel ritardare l’invio di un segmento finché non si accumula una quantità di dati sufficiente nel buffer. Non è possibile fissare un valore per il ritardo, perché TCP deve essere indipendente dall’applicazione e consentire una trasmissione efficiente qualunque sia la velocità di generazione del flusso.

Si usa un algoritmo adattativo (*Nagle algorithm*): non viene trasmesso un segmento finché non si hanno abbastanza dati da riempire un segmento di dimensione massima (MSS). Se la conferma del segmento precedente arriva prima, si trasmette il contenuto del buffer ossia tutto ciò che nel frattempo si è accumulato (questo garantisce che la applicazioni lente non aspettino troppo prima di vedere trasmessi i loro dati).

10.4.5 Il formato del segmento TCP

Ogni segmento è diviso in due parti, l’intestazione (header) e il campo dati (si veda la figura 10.8). L’intestazione TCP comprende:

- **source port e destination port:** contengono i TCP port number che individuano gli estremi della connessione;
- **sequence number:** è la posizione, nel flusso di byte del trasmettitore, del primo ottetto contenuto nel segmento;
- **acknowledgement number:** è il numero di sequenza del byte che il ricevitore si aspetta di ricevere (si riferisce al flusso di dati in senso opposto);
- **hlen:** specifica la lunghezza dell’header in multipli di 32 bit: è necessario poiché il campo options ha una dimensione variabile;
- **reserved:** è un campo riservato per usi futuri;

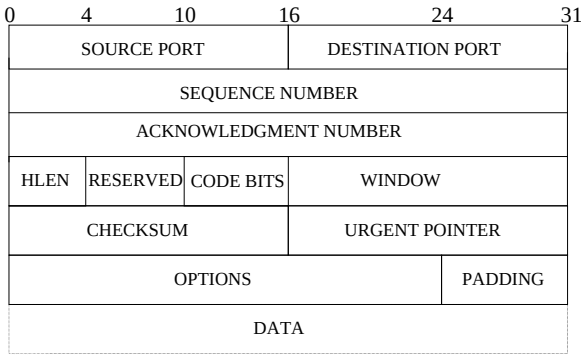


Figura 10.8. Formato del segmento TCP

- **code bits:** è usato per specificare il contenuto del segmento. Si usano i seguenti codici:

URG	il campo urgent pointer è valido
ACK	il campo acknowledgment è valido
PSH	questo segmento richiede una PUSH
RST	reset della connessione
SYN	sincronizza i numeri di sequenza
FIN	il trasmettitore ha raggiunto la fine dello stream

Il campo urgent pointer viene utilizzato quando alcuni dati devono essere trasmessi in modalità urgente, cioè senza rispettare la sequenzialità dei dati "normali". Il ricevitore deve consegnare quanto prima ai protocolli di livello superiore i dati urgenti, anche se altri dati dello stream giaciono ancora nel buffer di ricezione. Il meccanismo utilizzato per trattare i dati urgenti consiste nel marcare il bit URG e nello specificare nel campo urgent pointer la posizione dei dati all'interno del segmento. Il campo PSH viene utilizzato quando le applicazioni intendono forzare un trasferimento di dati. Il trasmettitore invia tutti i dati generati senza aspettare che il buffer sia pieno. TCP forza poi il ricevitore a renderli disponibili alle applicazioni senza alcun ritardo.

- **window:** specifica la dimensione corrente della finestra di ricezione che serve per il controllo di flusso;
- **urgent pointer:** specifica la posizione di eventuali dati urgenti all'interno del campo dati.

TCP: maximum segment size option

Un possibile uso del campo **options** è quello che consente agli applicativi interessati alla comunicazione di negoziare la MSS (Maximum Segment Size) cioè la massima dimensione del segmento che sono pronti a ricevere. Se un terminale ha uno spazio di memoria ridotto dedicato alla comunicazione, può richiedere la trasmissione di segmenti piccoli; se due utenti si trovano su una LAN possono scegliere di comunicare con dimensioni che consentono al segmento di rientrare in un singolo pacchetto del livello inferiore e ottimizzare così l'uso della banda.

La trasmissione di segmenti troppo piccoli risulta poco efficiente a causa dell'elevato rapporto tra informazione di controllo (intestazione) e informazione utile (dati). Segmenti troppo grandi sono frammentati su più IP datagram; questi ultimi non sono confermati o ritrasmessi indipendentemente: se non tutti i frammenti giungono a destinazione l'intero segmento viene perso. Un aumento della dimensione del MSS al di sopra della soglia di frammentazione diminuisce quindi la probabilità di successo della trasmissione. La scelta del MSS ottimale è condizionata da tre fattori:

- alcune implementazioni del protocollo TCP non danno la possibilità di gestire l'opzione;
- i cammini possono variare durante la connessione, per cui i pacchetti attraversano reti diverse con un MSS ottimo diverso;
- il valore ottinmo del MSS dipende dai protocolli di livello più basso con i quali non si ha comunicazione.

TCP: checksum computation

Il campo checksum nel TCP header contiene un intero su 16 bit usato per verificare l'integrità dei dati e delle informazioni di controllo contenute nello header. Per il calcolo del checksum si fa precedere il segmento da uno pseudo-header (in modo analogo a quanto visto per UDP), si allineano lo pseudo-header, l'intestazione TCP e i dati su blocchi da 16 bit e si calcola il complemento a uno della somma in complemento a uno dei blocchi. Nello pseudo-header, il campo TCP *length* specifica la lunghezza totale del segmento incluso l'header; *protocol* identifica il protocollo TCP.

10.4.6 Gestione delle connessioni in TCP

In questo paragrafo sono descritte le fasi di apertura e chiusura di una connessione.

La fase di apertura può essere effettuata seguendo due metodologie differenti dal punto di vista dell'applicativo:

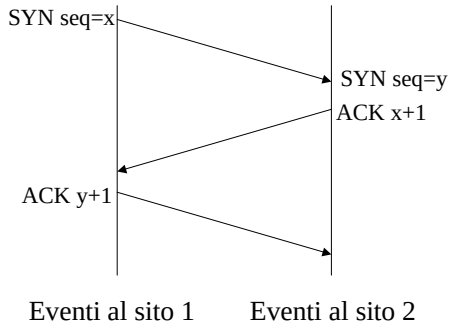


Figura 10.9. Procedura di apertura di una connessione in TCP - Three-Way Handshake

- **open passiva:** l'applicazione comunica che è disponibile ad accettare la comunicazione. Il sistema operativo assegna un port number; il socket remoto resta non specificato;

- **open attiva:** l'applicazione vuole comunicare con un processo remoto che aveva precedentemente fatto una open passiva: inizia la procedura di apertura di connessione detta *three-way handshake* (si veda la figura 10.9).

Il primo segmento di un handshake è identificabile per il SYN bit posto a 1. Nel secondo messaggio vengono settati i flag SYN e ACK per indicare che si tratta di una conferma alla richiesta di connessione. Il terzo segmento è un semplice ACK, usato per informare il destinatario che la connessione è stata stabilita.

È necessario usare tre messaggi per completare la procedura per evitare problemi nei casi in cui:

- entrambi gli host tentino la chiamata simultaneamente;
- la richiesta di apertura venga duplicata per la scadenza del timeout: il meccanismo a tre vie evita possibili ambiguità.

Durante la procedura di apertura i due processi si accordano sui numeri di sequenza iniziali: ciascun host sceglie in modo casuale il numero da cui partire per numerare il flusso di ottetti che trasferirà. L'host che inizia l'handshake comunica la sua cifra iniziale x mediante il campo sequenza del segmento SYN; il secondo utente, una volta ricevuto il segmento, registra tale valore e risponde con l'invio del proprio numero iniziale nel campo sequenza del secondo segmento (il cui campo acknowledgement specifica $x + 1$ come prossimo ottetto da ricevere). È possibile iniziare il trasferimento di dati già con il primo segmento di apertura: al ricevitore TCP li mantiene in memoria e li passa all'applicativo solo quando l'handshake è terminato.

Si noti che a tutti gli effetti il bit di SYN del primo segmento viene trattato come primo ottetto trasmesso: ad esso viene associato il numero di sequenza iniziale, e ad esso viene riferito il primo riscontro ricevuto. Lo stesso vale per il bit di FIN che discuteremo tra breve.

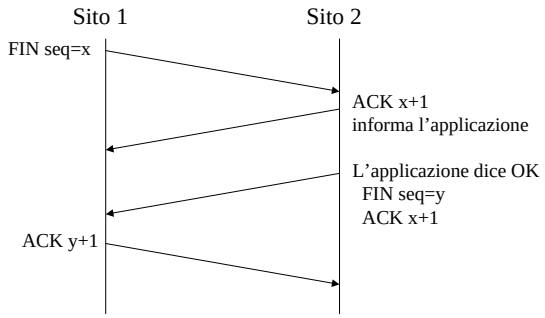


Figura 10.10. Procedura di chiusura di una connessione in TCP - full duplex

La chiusura di una connessione TCP (si veda la figura 10.10) prevede la chiusura in entrambe le direzioni vista la natura full-duplex. L'applicazione che non ha più dati da trasferire deve aspettare di aver ricevuto tutte le conferme prima di mandare un segmento con il flag FIN settato a 1. Al ricevitore, TCP manda un ACK e informa il processo che il flusso di dati è terminato. Se anche questa applicazione decide di terminare la comunicazione viene spedito un secondo segmento con il flag FIN e l'utente che ha inoltrato la richiesta di chiusura risponde con una conferma. Un meccanismo di questo genere si rende necessario perché non si conosce dopo quanto tempo il processo che ha ricevuto la richiesta di chiusura darà una risposta a TCP (ad esempio potrebbe attendere l'intervento dell'operatore). È quindi opportuno confermare subito il primo segmento per evitare ritrasmissioni.

A volte si verificano condizioni anomale che forzano un processo o il software di rete a chiudere la comunicazione; in questi casi TCP prevede la possibilità di un reset: all'invio di un segmento con il flag RST settato a 1 il destinatario risponde con la chiusura immediata della connessione (blocco del trasferimento dati e svuotamento dei buffer).

10.5 Il livello 3: IP Protocol (RFC 791)

Il protocollo IP (Internet Protocol) è il protocollo principale del livello 3 (Network) dell'architettura TCP/IP. Si tratta di un protocollo semplice che insieme a TCP costituisce il nucleo originale e principale dell'Internet Protocol Suite. Il servizio offerto può essere caratterizzato brevemente come:

- **inaffidabile:** la consegna del pacchetto non è garantita. I pacchetti possono essere persi, duplicati, ritardati o arrivare fuori sequenza senza che il servizio se ne accorga;
- **non connesso:** ogni pacchetto è trattato indipendentemente dagli altri. Una sequenza di pacchetti trasferita da un computer ad un altro può viaggiare lungo percorsi differenti;
- **best effort:** il protocollo non fa una selezione tra i pacchetti quando è in difficoltà. Cerca comunque di trasferire correttamente i messaggi ed è inaffidabile solo quando le risorse si esauriscono o si verifica un malfunzionamento di rete.

IP si occupa di instradare i messaggi sulla rete, ma ha anche funzioni di frammentazione e riassemblaggio dei messaggi e di rivelazione (non correzione) degli errori sull'intestazione.

10.5.1 Formato del pacchetto datagram IP

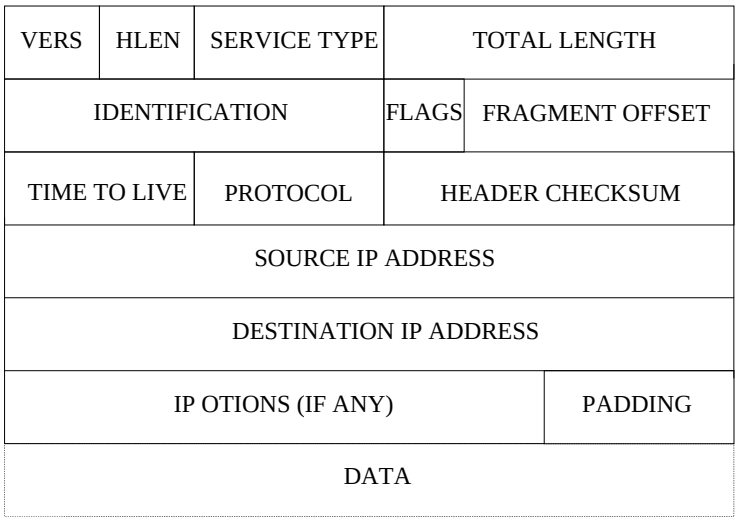


Figura 10.11. Formato del pacchetto IP

Il formato del pacchetto datagram IP è mostrato nella figura 10.11.

- **vers:** indica la versione di IP usata per creare il datagram; questo campo serve per verificare che sorgente, destinazione e ogni router intermedio concordino sul formato del pacchetto. La versione attuale del protocollo IP è la 4. È stata definita recentemente, ed è in fase di (lenta) installazione, la versione IP 6.

- **hlen:** specifica la lunghezza dell'header espressa in multipli di 32 bit; tutti i campi sono di dimensione fissata tranne le opzioni. L'header comunemente non contiene opzioni, per cui il valore del campo *hlen* è normalmente pari a 5.

- **total lengh:** specifica la lunghezza dell'intero datagram espressa in ottetti. Poiché si tratta di un intero su 16 bit, la massima dimensione del datagram IP è 65535 byte.

- **service type:** specifica come un protocollo di livello superiore vuole che il pacchetto sia trattato; è possibile assegnare vari livelli di priorità utilizzando questo campo; i primi tre bit del campo service type indicano il grado di precedenza del datagram che varia da 0 (precedenza normale) a 7 (controllo di rete). La maggior parte del software presente nei router ignora questa indicazione ma la sua presenza è concettualmente importante, perché consente alle informazioni di controllo di avere la precedenza sui dati: se i router supportassero le precedenze, si potrebbero implementare algoritmi di controllo di congestione insensibili alla congestione stessa e fornire supporto alla qualità di servizio. Gli ultimi due bit sono inutilizzati; i tre bit denominati **D, T, R** richiedono, se posti a uno, rispettivamente, bassi ritardi, alto throughput e alta affidabilità. Se il router ha la possibilità di scegliere tra diverse alternative di instradamento, i flag permettono di orientare la scelta sul cammino con le caratteristiche più appropriate.

- **identification:** è utilizzato per le operazioni di frammentazione del datagram, come i campi *fragment offset* e *flags*; contiene un intero che identifica l'IP datagram. Quando si rende necessario suddividere il datagram, l'IP header viene ricopiato in tutti i frammenti. Il destinatario, mediante il campo identification, sa a quale datagram appartengono i frammenti in arrivo.

- **fragment offset:** specifica l'offset dei dati contenuti nel frammento misurato in multipli di 8 ottetti.

- **flags:** due bit del campo flags sono significativi nelle operazioni di suddivisione. Il primo detto *do not fragment bit* indica, se posto a uno, che il datagram non può essere frammentato: se si presenta questa necessità il datagram è scartato. Il secondo bit (detto *no more fragments bit*) indica, quando posto a zero, che i dati contenuti nel frammento costituiscono l'ultimo frammento del datagram originale. Poiché i frammenti sono trasportati in rete in modo indipendente, e quindi possono arrivare fuori sequenza, mediante i campi *identification* e *fragment offset* il destinatario sa associare correttamente i frammenti e può metterli in ordine; il bit *no more fragments* permette di identificare la fine di un frammento.

- **time to live:** indica il tempo residuo di esistenza in rete del pacchetto. I router devono decrementare opportunamente questo campo ed eliminare il pacchetto dalla rete quando raggiunge il valore 0. Ogni volta che un router elabora un pacchetto, decrementa di uno il valore del campo; inoltre, per gestire i casi di congestione che causano alti ritardi, si decrementa il campo di una quantità pari al tempo in secondi per cui il pacchetto resta nella coda in attesa di trasmissione. Questo meccanismo impedisce che i datagram permangano nella rete per un tempo eccessivo, anche se si sono verificati malfunzionamenti nel processo di instradamento.

- **protocol:** indica quale protocollo di livello superiore ha creato (e per quale protocollo è destinato) il messaggio contenuto nell'area dati del datagram.

- **header checksum**: assicura l'integrità del solo header. L'header è allineato come una sequenza di interi a 16 bit. Il campo checksum contiene la somma in complemento a uno dei valori contenuti nell'header. Il principale vantaggio di disaccoppiare il controllo sui dati e sull'header è la riduzione dei tempi di elaborazione ai router. Il protocollo di livello superiore dovrà eventualmente adottare un controllo di correttezza sui dati.

- **source IP address e destination IP address**: contengono gli indirizzi IP dell'host sorgente e dell'host destinatario.

Concludiamo la descrizione dell'header IP con un cenno sulle opzioni previste dal protocollo. Il campo **IP options** permette di contenere informazioni opzionali, normalmente utilizzate per facilitare il controllo di funzionamento della rete. Ogni opzione consiste di un primo ottetto (*option code*) seguito dalla lunghezza dell'opzione e dai dati dell'opzione. Nel campo *option code* sono presenti, oltre al codice di identificazione dell'opzione *option number*, alcuni flag che indicano la classe dell'opzione (*option class*) e che ne richiedono la copia in tutti i frammenti di un datagram, oppure solamente nel primo (*copy flag*).

Le principali opzioni previste sono: record route, loose source route, strict source route e internet timestamp:

- **record route:** consente di registrare il percorso dei datagram: l'host sorgente crea una lista vuota di indirizzi IP che viene riempita dai router che instradano il pacchetto;

- **source route:** consente alla sorgente di specificare in modo più o meno preciso il percorso da seguire all'interno della rete Internet. La lista degli indirizzi contiene gli indirizzi dei router che il datagram dovrà attraversare: se l'opzione scelta è *strict source route* la lista dei router deve essere rispettata rigorosamente mentre la *loose source route* consente di attraversare router intermedi tra i router specificati;

- **timestamp:** consente ai router di registrare l'istante di tempo in cui viene elaborato il datagram.

10.5.2 Frammentazione/riassemblaggio di un IP datagram

La massima dimensione consentita per un datagram IP è 65535 byte. In realtà i frame trasportati dal livello fisico sono soggetti a vincoli più stringenti: spesso un IP datagram non può essere incapsulato in un unico frame, ma è necessario suddividerlo in più frammenti. L'operazione di frammentazione può avvenire in un qualunque punto del cammino: si rende necessaria quando un router riceve un pacchetto da una rete con una MTU (Maximum Transfer Unit) elevata e deve trasferirlo su una rete con MTU più piccola della dimensione del datagram. Da questo momento in poi i frammenti continuano ad essere trasportati senza essere riassemblati: l'operazione di riassemblaggio dei frammenti è effettuata solo dal ricevitore. All'operazione di frammentazione sono associati due aspetti negativi:

- una trasmissione non ottimale in termini di efficienza: ad ogni frammento generato deve essere copiata l'intestazione del segmento da cui è stato generato (molta informazione di controllo in rete);

- aumento della probabilità di scartare il datagram: la perdita di un solo frammento comporta la perdita dell'intero datagram.

Il grosso vantaggio di questa soluzione è la semplificazione del software dei router che, altrimenti, dovrebbero occuparsi di immagazzinare e riassemblare correttamente i frammenti con un incremento notevole del tempo di elaborazione.

10.5.3 Indirizzamento IP

L'indirizzamento IP è parte integrale del processo di instradamento dei messaggi sulla rete. Gli indirizzi IP, che devono essere univoci sulla rete, sono lunghi 32 bit (quattro byte) e sono espressi scrivendo i valori decimali di ciascun byte separati dal carattere punto. Esempi di indirizzi sono: 34.0.0.1, 129.130.7.4 e 197.67.12.3.

Agli indirizzi IP si associano per comodità uno o più nomi logici che possono essere definiti localmente in un file "host" che ha il seguente formato:

223.1.2.1	alpha
223.1.2.2	beta
223.1.2.3	gamma
223.1.2.4	delta
223.1.3.2	epsilon
223.1.4.2	iota

Questo approccio diviene impraticabile quando la rete IP cresce di dimensione e allora si preferisce utilizzare una base dati distribuita per la gestione dei nomi (DNS).

Gli indirizzi IP sono suddivisi in due parti. La prima parte indica l'indirizzo della rete **Net-id** (Network-identifier) e la seconda quello dell'host **Host-id** (Host-identifier).

Occorre subito evidenziare che non sono i nodi ad avere un indirizzo IP, bensì le interfacce verso le sottoreti. Quindi se un nodo ha tre interfacce, esso ha tre indirizzi IP. Poiché la maggior parte dei nodi ha una sola interfaccia, è comune parlare dell'indirizzo IP del nodo. Questo tuttavia è senza dubbio sbagliato nel caso dei router che hanno, per definizione, più di un'interfaccia. Gli indirizzi IP sono assegnati da un'unica autorità e quindi sono garantiti univoci a livello mondiale. Gli indirizzi IP sono stati originariamente suddivisi in cinque classi (si veda la figura 10.12):

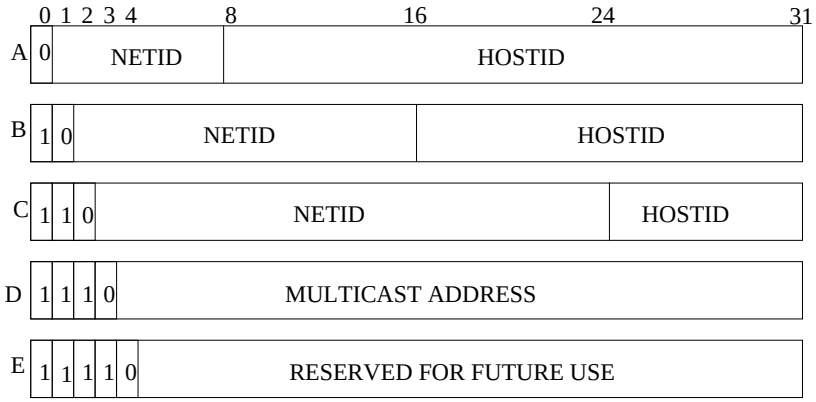


Figura 10.12. Classi di indirizzi di IP

- classe A: sono concepiti per poche reti di dimensione molto grandi. I bit che indicano la rete sono 7 e quelli che indicano l'host 24. Quindi si possono avere al massimo 128 reti di classe A, ciascuna con una dimensione massima di circa 16 milioni di indirizzi. Gli indirizzi di classe A sono riconoscibili in quanto il primo campo è compreso tra 0 e 127;

- classe B: sono concepiti per un numero medio di reti di dimensione medio grandi. I bit che indicano la rete sono 14 e quelli che indicano l'host 16. Quindi si possono avere al massimo 16304 reti di classe B, ciascuna con una dimensione massima di indirizzi pari a 65536. Gli indirizzi di classe B sono riconoscibili in quanto il primo campo dell'indirizzo è compreso tra 128 e 191;
- classe C: sono concepiti per moltissime reti di dimensione piccole. I bit che indicano la rete sono 21 e quelli che indicano l'host 8. Quindi si possono avere al massimo poco più di 2 milioni di reti di classe C, ciascuna con una dimensione massima di 256 indirizzi. Gli indirizzi di classe C sono riconoscibili in quanto il primo campo dell'indirizzo è compreso tra 192 e 223;
- classe D: sono riservati ad applicazioni di multicast secondo quanto descritto nello RFC 1112. Gli indirizzi di classe D sono riconoscibili in quanto il primo campo dell'indirizzo è compreso tra 224 e 239.
- classe E: questi indirizzi sono riservati per usi futuri. Gl indirizzi di classe E sono riconoscibili in quanto il primo campo dell'indirizzo è compreso tra 240 e 255.

Poiché tale struttura di indirizzi si è rivelata poco flessibile, essa è stata modificata nel tempo mediante l'uso delle subnet mask (discusso nello RFC 950) e del CIDR. Grazie a tali approcci, l'ampiezza dei campi net e host può essere definita in modo flessibile tramite un parametro detto **netmask**. Consideriamo inizialmente le subnet mask utilizzate principalmente per suddividere indirizzi di classe B in più sottoreti. La netmask contiene bit a uno in corrispondenza dei campi che identificano la rete e a zero in corrispondenza del campo host. All'intero di una sottorete IP la netmask deve essere univoca, in quanto il partizionamento della sottorete in subnet deve essere unico. La netmask viene messa in AND bit a bit con gli indirizzi IP per estrarre la parte network e subnet. Tramite questo procedimento è possibile verificare se due indirizzi appartengono alla stessa subnet. Ad esempio si supponga di avere una netmask 255.255.254.0 e i due indirizzi 128.155.4.77 e 128.155.5.75. Eseguendo l'AND bit a bit dei due indirizzi con la netmask si ottiene in entrambi i casi 128.155.4.0 e quindi gli indirizzi appartengono alla stessa subnet. Se consideriamo invece i due indirizzi 128.155.5.75 e 128.155.6.77 e la stessa netmask si ottengono dall'operazione di AND i due seguenti indirizzi 128.155.4.0 e 128.155.6.0, ovvero i due indirizzi appartengono a subnet differenti. La subnet mask è una soluzione locale che non modifica l'instradamento IP originario.

In origine la netmask veniva derivata implicitamente dalla **classe dell'indirizzo IP** considerato; questa soluzione, nata per facilitare l'instradamento, ha reso insufficiente il numero di indirizzi IP disponibili. Sono quindi stati imposti i **CIDR** (Classless Interdomain Routing). Questa nuova modalità di propagazione dell'informazione di instradamento tra router, associa ad ogni **indirizzo IP** una netmask. La grossa differenza rispetto ai protocolli non CIDR risiede nel fatto che anche indirizzi contigui vengono propagati tra router come fossero un indirizzo solo, operazione detta anche di clustering. Ad esempio, si supponga di voler far riferimento alle quattro seguenti reti di classe C: 199.9.4.0, 199.9.5.0, 199.9.6.0 e 199.9.7.0. Esse possono essere riferite contemporaneamente tramite l'indirizzo 199.9.4.0 e la netmask 255.255.252.0, che a tutti gli effetti possono costituire una sottorete definita al di là dello schema originale delle classi. In tal modo si riduce notevolmente la quantità di informazione che devono essere propagate dai vari protocolli di routing e quindi si aumenta l'efficienza. Inoltre, indirizzi contigui di classe C possono essere utilizzati anche in sottoreti comprendenti più di 256 host.

10.5.4 Associazione tra indirizzi IP e indirizzi fisici

Il problema dell'associazione tra l'indirizzo IP e l'indirizzo fisico è chiamato *address resolution problem* e si presenta ad ogni trasmissione di pacchetti tra host e router, router e router, router e host. Il protocollo IP prevede due tecniche:

1. **direct mapping**: se si ha la facoltà di scegliere sia l'indirizzo IP che quello fisico è possibile assegnare a quet'ultimo il valore del campo Host-id dell'IP address. Se gli indirizzi fisici non possono essere impostati liberamente, si ricorre a conversioni mediante tabelle.

2. **dinamic binding**: IP dispone di un meccanismo di address resolution efficace per reti come Ethernet, che sono intrinsecamente broadcast o possono implementare questa funzionalità facilmente. È possibile aggiungere liberamente nuovi host alla rete poiché viene eliminato il database centralizzato per il mapping che altrimenti dovrebbe essere aggiornato con i nuovi indirizzi. Questa tecnica è basata sui protocolli ARP (Address Resolution Protocol) e RARP (Reverse-ARP).

Entrambi i protocolli sono utilizzati per scoprire in modo automatico le corrispondenze tra gli indirizzi di livello 3 e gli indirizzi di livello 2 e viceversa. Questo è importante nelle LAN dove occorre creare una relazione tra gli indirizzi IP e gli indirizzi MAC.

Il protocollo ARP è utilizzato tutte le volte che una stazione collegata ad una LAN deve inviare un messaggio ad un nodo sulla stessa LAN (piú precisamente tra host con la stessa net-id) di cui conosce unicamente l'indirizzo di livello 3.

Il protocollo RARP viene invece utilizzato dalle stazioni non dotate di memoria di massa (diskless) per scoprire il loro indirizzo IP in fase di bootstrap.

Il meccanismo di base utilizzato da ARP prevede che, quando un host **A** vuole conoscere l'indirizzo fisico di un host o router **B**, trasmetta un pacchetto broadcast, detto di ARP request. Tutti gli host e i router collegati in rete ricevono il pacchetto, ma solo **B** riconosce il suo IP address e risponderà inviando un pacchetto diretto ad **A** contenente l'informazione richiesta, ovvero il suo indirizzo fisico.

Per rispondere, **B** ha bisogno dell'indirizzo fisico di **A**; quindi nell'ARP request **A** inserisce il suo IP address e il corrispondente indirizzo fisico. Ogni host registra in una cache le coppie di indirizzi (internet-fisico) degli altri host, ottenute:

- eseguendo la procedura descritta se si vuole comunicare con un host dall'indirizzo fisico ignoto;
- ogni volta che si riceve una ARP request (broadcast) si registra la coppia di indirizzi della sorgente anche se la request non riguarda l'host.

Nella figura 10.13 è presente il formato del pacchetto utilizzato dal protocollo ARP.
Il significato dei campi è il seguente:

- **operation**: è il tipo di operazione: *arp request*, *arp reply*, *rarp request* o *rarp reply*.
- **hardware type**: identifica i più importanti protocolli MAC (Ethernet=1).
- **protocol type**: identifica il protocollo che sta utilizzando ARP (IP=0800H).
- **hlen** e **plen**: permettono di utilizzare il protocollo ARP con rete arbitraria (indirizzi di dimensione variabile).

I protocolli ARP e RARP sono specificati nello RFC 826.

10.5.5 Il protocollo ICMP (Internet Control Message Protocol)

Internet Control Message Protocol (ICMP) è stato progettato per riportare anomalie che accadono nel routing di pacchetti IP e verificare lo stato della rete. ICMP è specificato nello RFC 792.

Nella tabella seguente sono riportati i tipi di pacchetti ICMP:

<table>
<tr><td colspan="2">Hardware Type</td><td>Protocol Type</td></tr>
<tr><td>HLEN</td><td>PLEN</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address (bytes 0-3)</td></tr>
<tr><td colspan="2">Sender Hardware Address (bytes 4-5)</td><td>Sender IP Address (bytes 0-1)</td></tr>
<tr><td colspan="2">Sender IP Address (bytes 2-3)</td><td>Target Hardware Address (bytes 0-1)</td></tr>
<tr><td colspan="3">Target Hardware Address (bytes 2-5)</td></tr>
<tr><td colspan="3">Target IP Address</td></tr>
</table>

Figura 10.13. Formato del pacchetto ARP

Type Field	Message Type
0	Echo Reply
3	Destination Unreachable
4	Source Quence
5	Redirect
8	Echo Request
11	Time Exceeded for a Datagram
12	Parameter Problem on a Datagram
13	Timestamp Request
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply

I messaggi che riportano anomalie sono incapsulati in datagram IP ed inviati agli host o router che hanno casusato l'errore; questi messaggi sono ad esempio, *destination unreachable*, *time exceeded for a datagram* e *parameter problem on a datagram*. I messaggi di verifica della raggiungibilità di un nodo sono *echo request* e *echo reply*.

Il messaggio *redirect* indica una condizione di stimolo ad un routing migliore, in quanto un router è stato attraversato inutilmente (ha dovuto ritrasmettere il messaggio sulla stessa rete da cui lo ha ricevuto).

Un router manda un messaggio ICMP di tipo *redirect* quando non è disposto ad instradare un pacchetto. Questo può avvenire o perchè il pacchetto può essere consegnato alla destinazione direttamente all'interno della sottorete, o per ridirigere il pacchetto verso un altro router. L'host in questo secondo caso associa un router diverso da quello di default per la destinazione contenuta nel pacchetto.

Gli ultimi messaggi ad essere stati introdotti nel protocollo ICMP sono *address mask request* e *address*

mask reply, per permettere ad una interfaccia di scoprire automaticamente la netmask usata in quella network.

10.6 Routing e relativi protocolli di instradamento

Con le subnet, si è introdotto il concetto di gerarchia su due livelli: un primo livello all'interno della subnet, implicito in quanto gestito dalla rete fisica; un secondo livello tra subnet gestito dagli IP router tramite tabelle di instradamento. Le subnet derivano dalla suddivisione di una network.

Il routing all'interno della subnet è banale in quanto la subnet coincide con una rete fisica che garantisce la raggiungibilità diretta delle stazioni ad essa collegate. Quando un pacchetto generato da un host è destinato ad un altro host appartenente alla stessa subnet si parla di **consegna diretta**. L'unico problema che si può incontrare è quello di mappare gli indirizzi IP nei corrispondenti indirizzi di livello 2. Questo mappaggio è oggi quasi sempre gestito in modo automatico, tramite i protocolli ARP e RARP, descritti nel paragrafo 1.5.4.

Il routing tra le subnet è gestito dagli IP router che effettuano l'instradamento sulla base di tabelle di instradamento che possono essere scritte o manualmente dal gestore della rete, se le reti sono piccole, o calcolate automaticamente tramite una serie di algoritmi di tipo distance-vector o link state packet. Quando un pacchetto generato da un host è destinato ad un altro host appartenente ad una differente subnet si parla di **consegna indiretta**. Per costruire le tabelle di instradamento, si utilizzano i vari algoritmi di routing. Un host o un router che deve trasmettere un datagram IP, esegue un'operazione di AND bit a bit tra l'indirizzo di destinazione del datagram e la netmask di ogni sua interfaccia; tale operazione consente di decidere se operare una consegna diretta, o una consegna indiretta attraverso un router.

Le network sono ulteriormente raggruppate in *Autonomous System* (AS), cioè in gruppi di network controllate e gestite da un'unica autorità amministrativa. Gli autonomous system sono identificati da un numero intero, univoco a livello mondiale, assegnato dalla stessa autorità che rilascia gli indirizzi Internet.

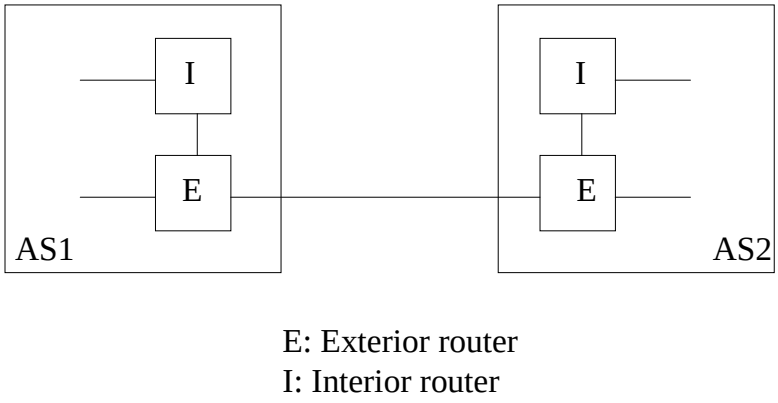


Figura 10.14. Interconnessione tra due AS differenti

I router che instradano messaggi all'interno dello stesso AS sono detti *interior router*, mentre quelli che instradano i messaggi anche tra AS diversi sono detti *exterior router*. Un esempio di interconnessione tra due AS differenti è mostrato in figura 10.14. Gli interior router possono scambiare informazioni di instradamento tramite un **IGP** (Interior Gateway Protocol), mentre gli exterior router utilizzano un **EGP** (Exterior Gateway Protocol).

10.6.1 Algoritmi di routing IGP

RIP (Routing Information Protocol)

RIP è un protocollo di tipo distance-vector in cui ogni router invia il suo distance vector ai router adiacenti ogni 30 secondi. Le tabelle di instradamento memorizzano un solo cammino per ogni destinazione. Il limite principale di RIP è che permette un numero massimo di hop pari a 15: ogni destinazione più lontana di 15 hop viene considerata non raggiungibile. Inoltre, RIP ignora le velocità delle linee, non permette di definire costi o altre metriche, ma basa l'instradamento solo sulla minimizzazione del numero di hop. In caso di modifiche della topologia della rete è un protocollo molto lento a convergere.

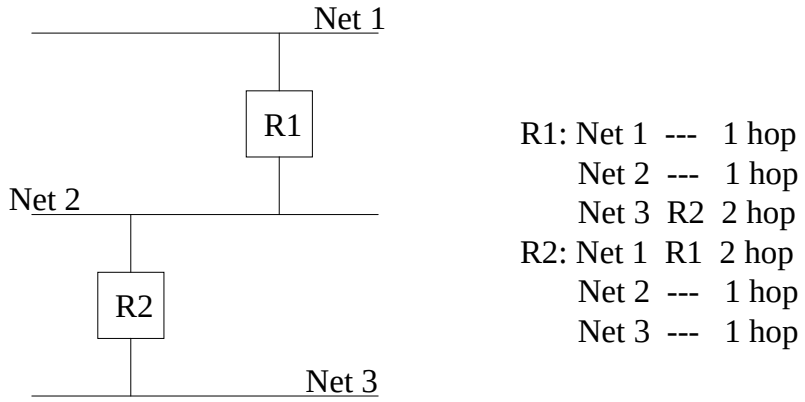


Figura 10.15. Problema del Count to Infinity

Questo algoritmo, essendo vincolato ad avere un solo cammino di uscita nella sua tabella di instradamento, soffre del problema del "Count To Infinity". Si consideri la figura 10.15. Si supponga che si venga a guastare il collegamento tra R1 e Net1; R1 aggiorna la sua tabella sostituendo, nella riga relativa alla Net1, costo 1 con costo infinito. Supponiamo però che R2 invia la propria tabella in cui Net1 risulta raggiungibile in 2 passi. R1 pensa in tal modo di poter raggiungere Net1 in 3 passi passando da R2 e aggiorna la sua tabella di instradamento inviandola a R2. Ma R2 raggiunge Net1 tramite R1 ed essendo cambiato il numero di hop necessari a R1 per arrivare a Net1 (ora sono 3), aggiorna la sua tabella settando il numero di hop a 4. Il procedimento porta l'algoritmo a divergere, ma il limite superiore di 15 hop arresta questa rapida crescita.

IGRP (Interior Gateway Routing Protocol)

IGRP è stato sviluppato da Cisco System a metà degli anni 1980 per superare i limiti di RIP.

Si tratta anche in questo caso di un protocollo di tipo distance-vector, ma con una metrica più sofisticata. La scelta del cammino migliore è effettuata da IGRP combinando vettori di metriche contenenti: ritardo, banda, affidabilità, lunghezza massima del pacchetto e carico. Inoltre IGRP permette il *multipath routing*, cioè la suddivisione del traffico tra più linee alternative. Il carico viene diviso in funzione delle metriche associate alle linee.

OSPF (Open Shortest Path First)

OSPF è stato sviluppato appositamente dall'IETF (Internet Engineering Task Force). Il gruppo di lavoro è stato costituito nel 1988 con lo scopo di realizzare un protocollo di tipo *link state packet* per TCP/IP. L'algoritmo prevede che ogni router disponga della mappa completa della rete su cui calcolare gli instradamenti ottimali utilizzando l'algoritmo di Dijkstra o *Shortest Path First.*

10.6.2 Algoritmi di routing EGP

EGP (Exterior Gateway Protocol)

EGP è il primo protocollo di tipo **EGP** ad essere stato ampliamente utilizzato nella rete Internet. Specificato con lo RFC 904 nel 1984 è oggi ampliamente disponibile su tutti i router, anche se è ormai considerato obsoleto e Internet lo sta sostituendo con il BGP.

EGP è un protocollo simile ad un algoritmo di distance vector, ma invece del concetto di costo specifica solo se la destinazione è raggiungibile oppure no.

I limiti di EGP somo molti e gravi: EGP non ha una metrica associata alle linee e quindi basa le sue decisioni esclusivamente sulla raggiungibilità; EGP non ammette la presenza di magliature nella topologia, e tutti gli AS devono essere collegati in modo stellare ad un core system.

BGP (Border Gateway Protocol)

BGP è un protocollo pensato per rimpiazzare il protocollo EGP. Il BGP è un algoritmo di tipo distance-vector, ma invece di trasmettere il costo di una destinazione, trasmette la sequenza di autonomous system da attraversare per raggiungere la destinazione. Ogni router calcola il suo instradamento preferito verso una data destinazione e lo comunica ai router BGP adiacenti tramite un distance vector.

10.7 Servizi principali forniti da Internet

I servizi forniti da Internet sono classificabili in:

- Servizi di Information Retrieval
  - FTP (File Transfer Protocol), trasferimento di file
  - Gopher
  - WWW (World Wide Web)
- Servizi di tipo Communication
  - telnet (emulazione di terminale)
  - e-mail (posta elettronica)
  - servizi multimediali (video e voce su Internet)
- Servizi di tipo Discussion
  - mailing list
  - USENET News

Altri servizi importanti, anche se non direttamente utilizzabili dagli utenti, sono:

- DNS (Domain Name Server);
- SNMP;
- X-Window.

Prima di analizzare i vari servizi sopra nominati, si deve far presente che la maggior parte delle applicazioni fornite su Internet sono organizzate sulla base dell'architettura client-server. Il server è il fornitore del servizio e il depositario dell'informazione cercata, mentre il client è costituito dall'interfaccia che permette all'utente di interagire con il server per fruire del servizio ed ottenere le informazioni. Il client richiede il trasferimento di informazioni, mentre il server è in ascolto per soddisfare le richieste dei client.

10.7.1 FTP File Transfer Protocol - RFC 959

Il servizio di trasferimento file è detto servizio ftp dal nome del protocollo utilizzato.

Questo applicativo permette ad un utente collegato ad un elaboratore di trasferire file da e verso un altro elaboratore; il trasferimento può essere cioè bidirezionale, permettendo così sia di prelevare che di depositare informazione. Si precisa che si usa una connessione di controllo (sulla porta 21) e una per il trasferimento dei file (sulla porta 20). La connessione deve essere aperta in modo esplicito e richiede all'utente di fornire uno username e una password validi sull'elaboratore remoto.

Per facilitare la reperibilità e l'accesso a dati pubblici, è possibile creare una connessione anonima su alcuni server che forniscono tale servizio. La connessione anonima su un sito non richiede l'uso di una vera password, ma spesso è richiesta come password l'indirizzo di e-mail. L'accesso è normalmente fornito in sola lettura, talvolta ad un numero limitato di utenti e in particolari orari.

I comandi fondamentali utilizzabili con ftp sono:

Comandi	Effetto
cd	cambio di direttorio sul server
dir o ls	elenco file presenti sul server
get	prelevamento dal server di un file singolo
mget	prelevamento dal server di un insieme di file (uso di wildcard)
put	trasferimento sul server di un file singolo
mput	trasferimento sul server di un insieme di file (uso di wildcard)
bin/ascii	selezione della modalità di trasferimento
lcd	cambio direttorio locale
bye/exit	uscita dal programma

Spesso i file sono compressi per ovvi motivi di riduzione dello spazio necessario sul server ftp. I file compressi sono identificati mediante la loro estensione:

- l'estensione **.zip** si riferisce alla compressione effettuata utilizzando il programma **pkzip**, il più diffuso compressore su MS-DOS.
- l'estensione .**Z** si riferisce a programmi compressi utilizzando il programma **compress** caratteristico dei sistemi Unix.

- l'estensione **.gz** si riferisce a programmi compressi utilizzando il programma **gzip** caratteristico dei sistemi Unix.
- l'estensione **.tar** si riferisce ad un insieme di programmi non compressi, ma creati utilizzando il programma di backup di Unix.

Chi preleva file compressi dovrà naturalmente decomprimerli con il programma opportuno prima di poterli utilizzare. I file compressi devono ovviamente essere trasferiti in modalità binaria.

Ricerca di file su sito ftp: Archie

Archie è uno strumento di ricerca che permette di individuare il sito anonimo su cui sono reperibili file. Esistono numerosi server archie, ciascuno dei quali indicizza preferibilmente server ftp geograficamente vicini. Il database utilizzato per le ricerche su ogni server è aggiornato periodicamente.

Ad esempio, il comando **archie -h archie.sura.net -s wintool** avvia una ricerca sull'archie server **archie.sura.net**, per individuare tutti i siti ftp anonimi su cui si trovano file o direttori che contengono la stringa "wintool" nel nome.

10.7.2 Gopher: servizio di navigazione a menu

Gopher è il primo programma per la gestione di informazioni di tipo testuale introdotto su Internet; è stato sviluppato all'Università di Minnesota nel 1992. È il primo esempio di programma in cui gli utenti possono diventare **creatori di informazione**, non solo consumatori. Permette il reperimento di informazione di tipo testuale (ma anche di immagini, suoni e programmi eseguibili) basandosi su una struttura guidata a **menu**. Il tipo di interazione è del tutto analogo a quello che si è poi sviluppato per il WWW: si accede ad un server che fornisce il servizio gopher e si inizia un processo di ricerca dell'informazione.

Il processo di ricerca dell'informazione è però guidato attraverso una struttura rigida a menu. La sua diffusione è stata minima a causa dell'introduzione da parte del CERN del WWW, la cui diffusione ha immediatamente seguito e ben presto di gran lunga superato quella del gopher.

Veronica: ricerca sui gopher

Veronica svolge per il servizio gopher una funzione simile a quella svolta da archie per il servizio ftp.

La ricerca avviene per corrispondenza tra la parola chiave ricercata e il titolo del menu. È possibile utilizzare operatori logici per la ricerca. L'accesso avviene attraverso una voce di menu del gopher.

10.7.3 Il WWW (World Wide Web)

Il WWW, introdotto da studiosi del Cern di Ginevra nel 1990 e presentato nel 1991, ha avuto una diffusione enorme ed è stata la causa principale dell'esplosione recente di Internet (si calcolano tassi di crescita del 400% annuo come numero di utenti collegati). Permette la consultazione di documenti disponibili su Internet in modalità ipertestuale. Un **ipertesto** è un testo in cui è possibile associare ad alcuni elementi (ad esempio le parole) un riferimento ad altri documenti. Tali parole sono dette *ancore* o *collegamenti ipertestuali*. Fornisce una interfaccia grafica semplice e intuitiva, che contiene testi, immagini, filmati e che consente di creare interfacce per l'accesso a basi di dati e archivi. Permette una compabilità semplice con i servizi di altro tipo forniti su Internet quali news, ftp, telnet, gopher. Si basa sul protocollo **HTTP** (Hyper Text Transfer Protocol), definito negli RFC 1945 (versione 1.0) e RFC 2068 (versione 1.1), per lo scambio di messaggi tra il server WWW ed il client WWW. Le pagine sono descritte utilizzando il linguaggio **HTML** (Hyper Text Markup Language), un linguaggio per la descrizione degli ipertesti.

Come funziona il WWW?

L'utente (il client) richiede una risorsa specificandone la **URL** (Uniform Resource Locator) mediante il browser (visualizzatori dei documenti WEB). La URL non necessariamente si riferisce ad una pagina WWW, ma ad una generica risorsa Internet. Il browser intrepreta la URL ed inoltra una richiesta al server opportuno utilizzando il protocollo appropriato (FTP, GOPHER, HTTP). Il server fornisce la risorsa richiesta (oppure un messaggio di errore) utilizzando lo stesso protocollo. Il browser interpreta i contenuti del messaggio di risposta ed agisce di conseguenza. Se la risorsa è una pagina WWW, ovvero un documento HTML, il browser automaticamente inoltra la richiesta delle informazioni necessarie a completare la visualizzazione del documento (filmati, immagini, applet java, ecc.). Il browser interpreta la risorsa ricevuta e la presenta sullo schermo eventualmente con l'ausilio di programmi esterni per la presentazione di filmati o di audio di qualità; tali programmi sono detti plug-in. I client WWW, i browser, sono quindi programmi che interpretano il linguaggio HTML e presentano sullo schermo le pagine WWW e che utilizzano per il trasferimento delle informazioni il protocollo HTTP e il formato di indirizzamento proprio delle URL. I browser sono anche in grado di utilizzare altri protocolli per fornire interfaccia con i servizi più tradizionali disponibili su Internet.

Per il funzionamento del WWW, oltre gli standard presenti nella rete Internet necessari per fornire i servizi più tradizionali, sono necessari i seguenti standard:

- il formato MIME (RFC 2045, 2046, 2047 e 2048) per la definizione del formato dei dati, estensione del formato dei messaggi di posta elettronica definiti nello RFC 822;
- il formato URL (RFC 2396) per la definizione del formato utilizzato per identificare le risorse;
- il protocollo HTTP (RFC 2068) per la trasmissione delle informazioni tra client e server WWW;
- il linguaggio HTML per la descrizione dei documenti ipertestuali;
- l'interfaccia CGI per potere dialogare con altre applicazioni non appartenenti al mondo WWW (ad esempio, per ottenere accesso alle basi dati).

URL: Uniform Resource Locator

Per identificare una risorsa su Internet si utilizza una forma di indirizzamento definita in modo formale mediante una sintassi; la risorsa è denominata URL (Uniform Resource Locator) oppure URI (Uniform Resource Identifier). Il formato è definito nello RFC 2396.

Una URL definisce:

- un protocollo per accedere alla risorsa (ovvero quale servizio contattare);
- opzionalmente, una coppia $\langle$nome_utente,password$\rangle$, ad esempio per il servizio ftp;
- il nome del server (un indirizzo IP simbolico o numerico) presso cui il servizio è disponibile ed opzionalmente la porta TCP a cui tale servizio viene fornito;
- il direttorio (path) che contiene la risorsa;
- il nome della risorsa (nome file che la contiene);
- eventuali parametri che si intende passare alla risorsa.

Ad esempio, nella URL http://hp0tlc.polito.it/bianco/risorsa.html, si possono distinguere:

- il protocollo *http*;

- il server *hp0tlc.polito.it*;
- il path *bianco* come il direttorio in cui trovare la risorsa cercata.

10.7.4 Il servizio Telnet

Il servizio Telnet permette di accedere in modo remoto ad host collegati alla rete Internet. Un utente è in grado di operare su un host remoto come se fosse direttamente collegato ad esso.

Telnet crea una emulazione di terminale: per questo motivo è necessario specificare di quale terminale si vuole eseguire l'emulazione. Il programma telnet emula il protocollo di comunicazione tra terminale e host. È possibile specificare a quale porta ci si vuole collegare; se si sceglie una porta diversa dalla porta che fornisce il servizio telnet si può entrare in comunicazione con altri processi che forniscono servizi quali la posta elettronica o un servizio http e "dialogare" con questi processi utilizzando l'opportuno protocollo.

10.7.5 E-mail, il servizio di posta elettronica

Il servizio di posta elettronica permette di trasferire messaggi tra due utenti della rete Internet.

Gli applicativi di posta elettronica sono specifici dei sistemi operativi: tutti permettono di ricevere e spedire messaggi, di gestire elenchi di indirizzi e gruppi di indirizzi per tematiche, di specificare il Subject, ovvero l'argomento del messaggio, e di inviare il messaggio in copia ad altri utenti.

Un indirizzo di posta elettronica è caratterizzato da un *nome utente*, un simbolo "*@*" e *l'indirizzo simbolico di un host* (bianco@hp0tlc.polito.it).

Questa scelta degli identificatori di utenti del servizio di posta elettronica presenta alcuni svantaggi:

- Il nome utente si riferisce allo username, o login name, o in generale alla stringa di caratteri che si utilizza quando si accede all'host; spesso tale stringa è solo lontana parente del cognome di una persona e talvolta non ha nessuna relazione con il cognome.
- Inoltre, nel momento in cui l'utente utilizza un nuovo host, con un indirizzo simbolico differente, è necessario modificare l'indirizzo di posta elettronica per poterlo raggiungere.

Per ovviare a questo problema, alcune organizzazioni forniscono un servizio di traduzione di posta elettronica. Un utente sarà identificato dall'esterno non più utilizzando il nome utente e host ma con il suo nome, cognome e il nome dell'organizzazione a cui è legato; ad esempio: andrea.bianco@polito.it. Sarà compito di un programma di traduzione effettuare questa trasformazione di indirizzo. In questo modo un cambiamento di nome utente o di host non richiede modifiche all'indirizzo di posta ma solo al data base che traduce localmente gli indirizzi.

Il servizio di posta elettronica permette di trasferire messaggi e può quindi essere utilizzato anche per il trasferimento di file in caratteri ASCII, che comprendono file testo, file html e file postscript. Essendo un mezzo molto comodo per trasferire file, che non richiede la conoscenza della password del destinatario, gli utenti utilizzano spesso programmi che codificano file di tipo binario (immagini, filmati, programmi eseguibili, file compressi) in formato testo. In questo modo si possono trasferire file di qualsiasi formato. Poiché i programmi di codifica espandono la dimensione dei file, è buona norma prima comprimere i file e poi codificarli, anche perché molti server di posta non accettano messaggi di dimensioni eccessive. Chi riceve il messaggio dovrà eseguire il processo di decodifica opportuno per poter utilizzare il file.

Purtroppo non esiste una guida degli indirizzi di posta elettronica, se per guida si intende una guida analoga a quella telefonica, disponibile in tutti i paesi del mondo e che comprende un elenco aggiornato di tutti gli utenti. Sono disponibili svariati siti WWW che permettono di ottenere informazioni di traduzione tra nomi di persone ed indirizzi di posta elettronica:

- Internet Address Finder, all'indirizzo www.iaf.net;
- Infoseek, www.infoseek.com;
- AOL Netfind, www.aol.com/netfind;
- HotBot, www.hotbot.com;
- Yahoo, www.yahoo.com;
- WebCrawler, www.whowhere.com/WebCrawler/wc_search.html.

Il protocollo utilizzato per il trasferimento di messaggi di posta elettronica è **SMTP** (Simple Mail Transfer Protocol), definito nello RFC 821 e nello RFC 822. In questo protocollo, ogni utente è identificato dalla sintassi *Utente@Elaboratore* e non è richiesta alcuna autorizzazione per poter inviare un messaggio di posta elettronica. Il procedimento avviene in modo batch, riprovando più volte sino a quando l'elaboratore remoto non diventa raggiungibile. L'utente remoto viene avvisato dell'arrivo di un nuovo messaggio.

10.7.6 Mailing list

Le mailing list sono "forum di discussione" che utilizzano la posta elettronica. Esistono migliaia di mailing list sui più svariati argomenti. Alcune mailing list sono moderate, altre non moderate.

Per iscriversi ad una mailing list è necessario spedire un messaggio di posta elettronica ad un opportuno indirizzo, contenente la parola subscribe nel testo oppure nell'argomento del mail (Subject).

Per inviare un messaggio si utilizza un indirizzo specifico per la mailing list. Tale messaggio sarà automaticamente inviato a tutti gli iscritti alla mailing list, eventualmente previo controllo del contenuto da parte del moderatore.

Di norma gli indirizzi a cui inviare i messaggi e a cui iscriversi alla lista sono diversi tra di loro.

10.7.7 News: Gruppi di discussione o newsgroup

Sono un forum di discussione organizzato in maniera diversa rispetto alle mailing list. Si avvicinano come gestione al concetto di bacheca.

Esistono una quantità inimmaginabile di newsgroup, sui più svariati argomenti. Gli argomenti sono organizzati in categorie:

Categoria	Arg. di discussione	Num. di gruppi	Esempio
news	info generali sui newsgroup	30	news.answers
comp	info sui calcolatori	900	comp.database
rec	arg. di varia natura, cinema, libri ..	696	rec.birds rec.humor
sci	natura scientifica	203	sci.agriculture
soc	sociologia, cult. nazionale	265	soc.culture.italian
alt	alternativi	2184	alt.binaries.picture
it	gruppi italiani	282	it.notizie

È possibile consultare i messaggi spediti al newsgroup accedendo ad uno tra i numerosi server che si scambiano i messaggi ricevuti, mantenendo una copia di tutti i messaggi ricevuti. È possibile anche rispondere a messaggi specifici o "postarne" di nuovi.

Passato un periodo di tempo normalmente pari a tre settimane (ma dipende dal server e dal newsgroup!), i messaggi sono automaticamente eliminati dal server e non sono più disponibili.

Esistono programmi specifici che permettono di accedere ai newsgroup, ma i browser più diffusi (i programmi che permettono di consultare le pagine HTML del WWW) offrono la stessa possibilità.

La creazione di un newsgroup segue regole precise, passando attraverso una votazione di approvazione. Esistono alcune regole di "buona educazione", la cosidetta netiquette, che è bene seguire per accedere ai newsgroup:

- leggere le FAQ (Frequently Asked Questions) prima di spedire messaggi;
- inviare messaggi brevi e chiari;
- verificare che il gruppo a cui si spedisce il messaggio sia quello appropriato;
- non inviare messaggi a troppi gruppi contemporaneamente (crossposting);
- MAI SCRIVERE IN MAIUSCOLO (equivale ad urlare);
- evitare file di signature (le firme) enormi.

Il protocollo utilizzato per trasferire i messaggi tra i server che permettono l'accesso ai newsgroup è **NNTP**, definito nello RFC 977.

10.7.8 DNS-Domain Name Server

Per evitare di utilizzare gli indirizzi numerici per identificare un host, ad ogni indirizzo numerico è associato un indirizzo **simbolico**, costituito da una sequenza di nomi, separati dal carattere "." Ad esempio: pol88a.polito.it. Il nome è assegnato in modo da permettere di riconoscere la locazione e/o la natura dell'host. Gli indirizzi simbolici sono organizzati in un **dominio principale** e in domini locali o sottodomini. I domini principali sono identificabili dal nome con cui termina l'indirizzo simbolico, i domini locali precedono direttamente i domini nell'indirizzo simbolico.

Dominio	Identifica
.it .uk .fr	Sito non statunitense (codici nazionali)
.com	Organizzazione commerciale
.org	Organizzazioni non commerciali
.gov .mil	Sito governativo e militare
.net	Fornitori di servizi di rete

Nell'indirizzo simbolico pol88a.polito.it, it è il dominio, polito è un sottodominio, pol88a è l'host.

Si devono precisare due importanti concetti su DNS:

- l'assoluta indipendenza tra nomi ed indirizzi;
- la presenza di una gerarchia di server DNS.

10.7.9 SNMP-Simple Network Management Protocol

SNMP è un protocollo per la gestione degli apparati, basato su UDP/IP. SNMP è stato progettato per inviare dati sullo stato della rete provenienti dagli apparati ad un centro di gestione che li interpreti in modo opportuno. Con SNMP è anche possibile modificare alcuni parametri degli apparati di rete.

10.7.10 X-Window

X-Window è un software di rete client-server che permette ad un programma client di visualizzare dati grafici del display di un altro elaboratore che funge da server grafico. Nato nell'ambito del progetto MIT Athena, X-Window si è diffuso su tutti gli elaboratori e su tutti i protocolli, tra cui anche TCP/IP.