

# Reti

Giuseppe Facchi

# Indice

<b>1</b>	<b>Livello di collegamento e reti locali</b>	<b>3</b>
1.1	Introduzione . . . . .	3
1.1.1	Servizi offerti dal livello di collegamento . . . . .	3
1.1.2	Dov'è implementato il livello di collegamento? . . . . .	3
1.2	Tecniche di rilevazione e correzione degli errori . . . . .	4
1.2.1	Controllo di parità . . . . .	5
1.2.2	Checksum ( <i>solo</i> strato di trasporto) . . . . .	7
1.2.3	CRC (Controllo a rindondanza ciclica) . . . . .	7
1.3	Collegamenti broadcast e protocolli di accesso multiplo . . . . .	9
1.3.1	Protocolli a suddivisione del canale . . . . .	10
1.3.2	Protocolli ad accesso casuale . . . . .	12

# 1 Livello di collegamento e reti locali

## 1.1 Introduzione

- **Nodo:** Generico terminale
- **Link:** Collegamento tra terminali

Su ogni collegamento, un nodo trasmittente incapsula il datagramma in un **frame di livello di collegamento** *link-layer frame* e lo trasmette lungo il collegamento stesso

### 1.1.1 Servizi offerti dal livello di collegamento

- **Framing:** Incapsulamento datagrammi del livello di rete all'interno di un frame a livello di collegamento
- **Accesso al collegamento:** Protocollo MAC *medium control access*, il quale specifica le regole con cui immettere i frame nel collegamento
- **Consegna affidabile**
- **Rilevazione e correzione dell'errore:** Il nodo ricevente può erroneamente decidere che un bit in un frame sia 0 quando era stato trasmesso come 1, e viceversa. Gli errori sui bit sono causati da *attenuazione del segnale* e da *disturbi elettromagnetici*

### 1.1.2 Dov'è implementato il livello di collegamento?

Per un dato collegamento, il protocollo del livello di collegamento è realizzato da un **adattatore di rete** *network adapter*, noto anche come **scheda di rete** *network interface card*.

La maggior parte delle funzionalità del controller è implementata a *livello hardware*

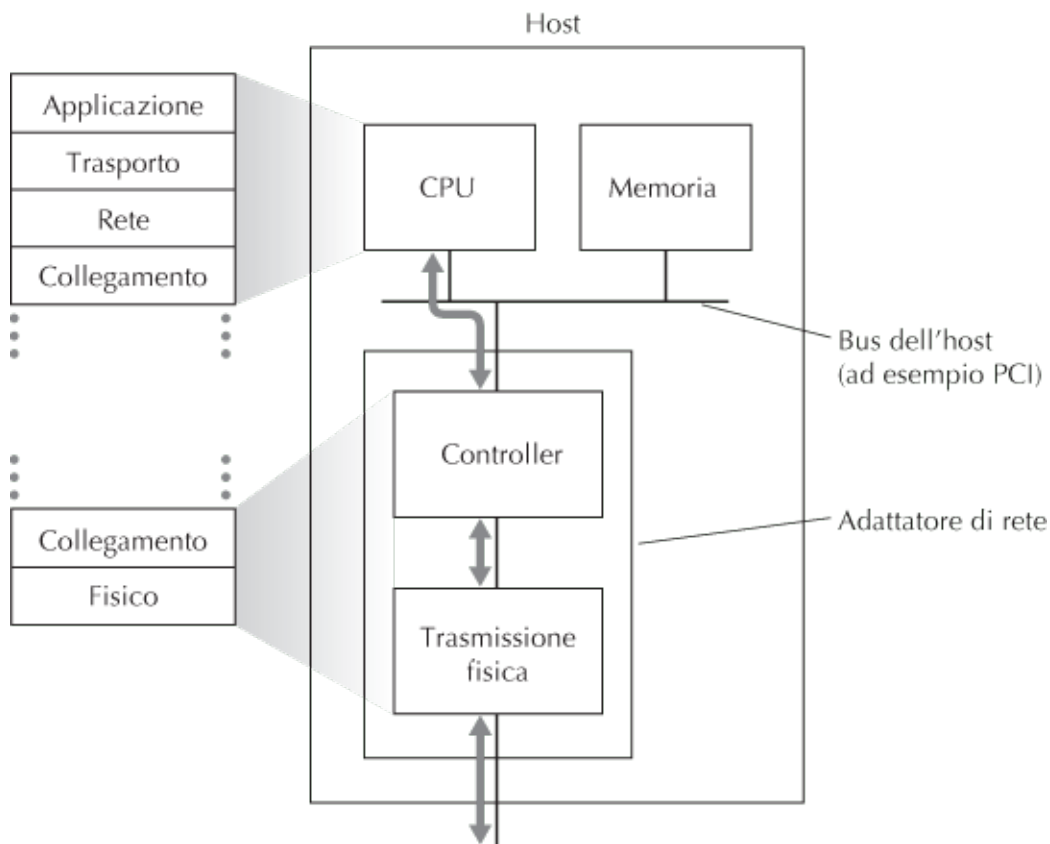


Figura 1: Scenario di rilevazione e correzione errori

## 1.2 Tecniche di rilevazione e correzione degli errori

Al nodo trasmittente, ai dati  $D$  che devono essere protetti da errori vengono aggiunti dei bit detti *EDC* (*error detection and correction*).

I dati  $D$  e i bit *EDC* sono inviati in un frame al nodo ricevente. Questo legge una sequenza di bit  $D'$  ed  $EDC'$  che può essere diversa dall'originale, come risultato della modifica dei bit in transito.

**Il nodo ricevente deve determinare se  $D'$  coincida con  $D$ .**

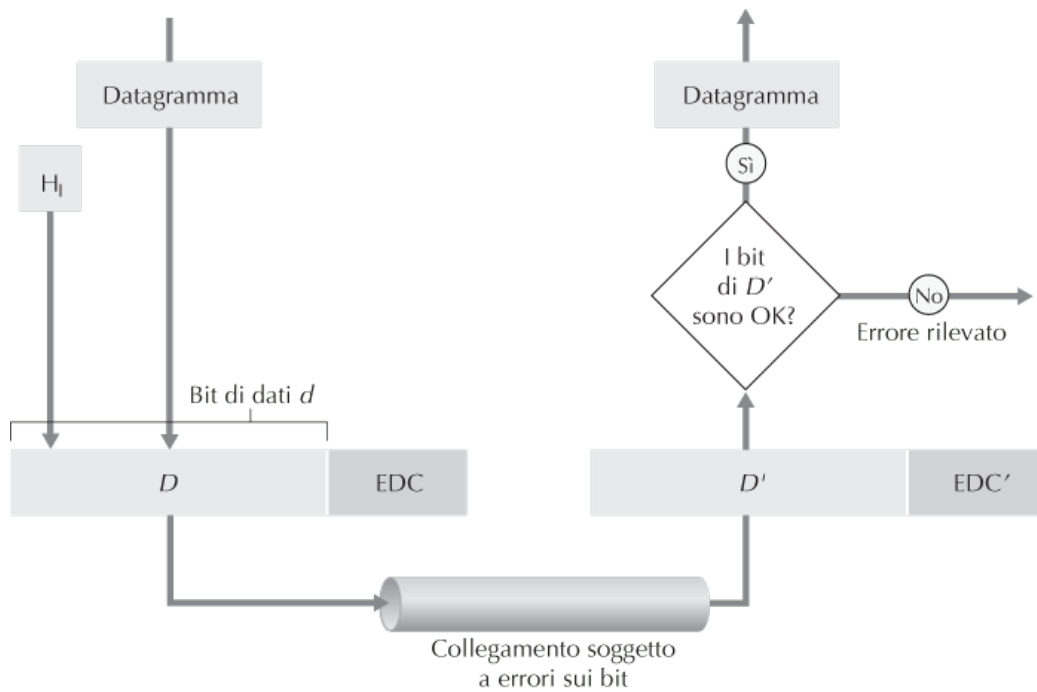


Figura 2: Scenario di rilevazione e correzione degli errori

Anche con l'utilizzo dei bit di rilevazione degli errori è possibile che ci siano degli **errori non rilevati**.

**Tecniche rilevazione degli errori:**

- Controllo di parità
- Tecniche di checksum
- Controllo a rindondanza ciclica

### 1.2.1 Controllo di parità

La forma più semplice di rilevamento degli errori è quella che utilizza un **unico bit di parità** (*parity bit*).



Figura 3: Parità "pari" a un bit

**Mittente:**

***Schema pari***

- Include un bit aggiuntivo
- Sceglie il suo valore in modo da rendere pari il numero totale di bit a 1 nei  $d + 1$  bit trasmessi

***Schema dispari***

- Include un bit aggiuntivo
- Sceglie il suo valore in modo da rendere dispari il numero totale di bit a 1 nei  $d + 1$  bit trasmessi

**Destinatario:**

- Conta il numero di bit a 1 tra quelli ricevuti
- Se trova un numero dispari di bit 1, sa che si sono verificati *un numero dispari di errori*

**Numero pari di errori nei bit** *Errore non rilevato.*

**Parità Bidimensionale:** I  $d$  bit del dato  $D$  sono suddivisi in  $i$  righe e  $j$  colonne per ognuna delle quali è stato calcolato un valore di parità. I risultanti  $i + j + 1$  bit di parità contengono bit per la rilevazione dell'errore

Il ricevente può utilizzare l'indice di riga e colonna per individuare il bit alterato.

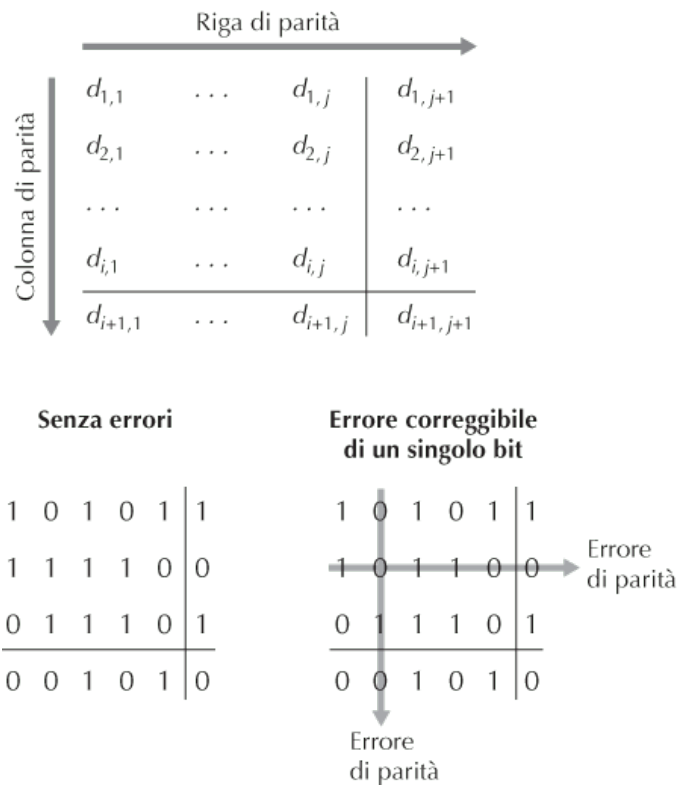


Figura 4: Parità pari bidimensionale

**Forward error correction (FEC):** Capacità del ricevente sia di rilevare sia di correggere gli errori

### 1.2.2 Checksum (*solo strato di trasporto*)

Nelle tecniche che utilizzano il checksum i  $d$  bit di dati sono trattati come interi da  $k$  bit.

**Checksum di Internet:** I dati sono trattati come interi di 16 bit e sommati. Il complemento a 1 di questa somma costituisce il checksum.

### 1.2.3 CRC (Controllo a rindondanza ciclica)

Una tecnica di rilevazione dell'errore largamente utilizzata nelle più recenti reti di calcolatori è basata sui **codici di controllo a rindondanza ciclica**

(CRC). I codici CRC sono anche detti **codici polinomiali**.

**Codice polinomiale:** Rappresentazione di una generica stringa di bit da trasmettere come un **polinomio** i cui coefficienti sono i bit della stringa, con le operazioni sulla stringa di bit interpretate come aritmetica polinomiale.

- $D$ : Dati visti come numero binario
- $G$ : Generatore ( $r + 1$  bit scelti)
- $R$ : Bit scelti in modo che  $D \cdot R$  esattamente divisibile per  $G$  (conosciuto dal receiver)

Se il resto è **non-zero** viene rilevato un errore

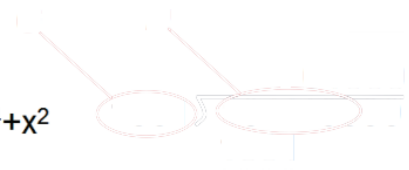
$D=100101$

$G=101$  ( $r=2$ )

$D \cdot 2^r=10010100$

$\rightarrow x^2+1$

$\rightarrow x^7+x^4+x^2$



$$\begin{array}{r}
 x^7+x^4+x^2 \\
 x^7+x^5 \\
 \hline
 x^5+x^4+x^2 \\
 x^5+x^3 \\
 \hline
 x^4+x^3+x^2 \\
 x^4+x^2 \\
 \hline
 x^3 \\
 x^3+x \\
 \hline
 x
 \end{array}$$

$\left| \begin{array}{r} x^2+1 \\ \hline x^5+x^3+x^2+x \end{array} \right.$

Con il potente aiuto della aritmetica dei polinomi a coefficienti in GF(2)!

Trasmetto 10010110

$\rightarrow x^7+x^4+x^2+x$  divisibile per  $x^2+1$

→ 10=R

Figura 5: Esempio di CRC



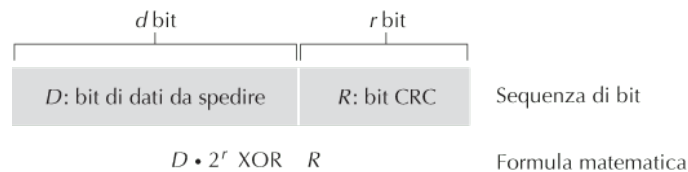


Figura 6: Esempio di CRC

### 1.3 Collegamenti broadcast e protocolli di accesso multiplo

Due tipi di collegamento di rete:

- **Collegamento punto a punto:** È costituito da un trasmittente a un'estremità del collegamento e da un unico ricevitore.
- **Collegamento broadcast:** Può avere più nodi trasmittenti e riceventi connessi allo stesso canale broadcast condiviso.

**Problema dell'accesso multiplo:** Come coordinare l'accesso di più nodi trasmittenti e riceventi in un canale di broadcast condiviso.

**Protocolli ad accesso multiplo:** Fissano le modalità con cui i nodi regolano le loro trasmissioni sul canale condiviso.

- Protocolli a suddivisione del canale
- Protocolli ad accesso casuale
- Protocolli a rotazione

**Collisione:** Se due o più nodi trasmettono un frame nello stesso istante e nello stesso canale si verifica una **collisione**, provocando nei riceventi una *non comprensione* dei dati appena ricevuti.

**Protocollo di accesso multiplo ideale:** Canale broadcast di tasso  $R$  bps

- Quando un solo nodo vuole trasmettere, trasmette a  $R$  bps
- Quando  $M$  nodi vogliono trasmettere, ognuno in media può inviare ad un tasso di  $\frac{R}{M}$  bps

- Completamente decentralizzato
  - Nessun nodo speciale per coordinare trasmissioni
  - No clock di sincronizzazione
- Semplice

### 1.3.1 Protocolli a suddivisione del canale

**Divisione di canale: TDMA** (*Time Division Multiple Access*)

- Accesso al canale periodico
- Ogni stazione ottiene slot di lunghezza fissata ( $L = RTT$ ) ad ogni giro
- Slot inutilizzati vanno in idle
- **Evita le collisioni**

**Esempio:**

- 6 stazioni su LAN
- 1, 3, 4 hanno pacchetti
- 2, 5, 6 sono in idle



Figura 7: TDMA

**Divisione di canale: FDMA** (*Frequency Division Multiple Access*)

- Spettro del canale diviso in bande di frequenza
- Ad ogni stazione assegnata banda fissata di frequenza
- Tempo inutilizzato per la trasmissione nella banda di frequenza è idle
- **Evita le collisioni**

**Esempio:**

- 6 stazioni su LAN
- 1, 3, 4 hanno pacchetti
- Bande di frequenza di 2, 5, 6 sono in idle

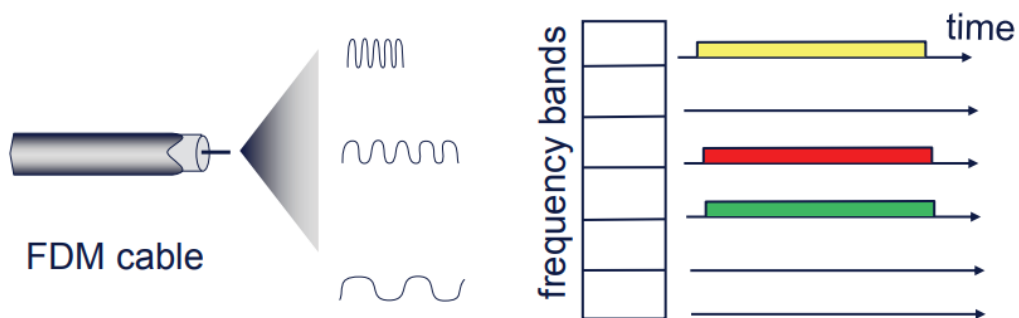


Figura 8: FDMA

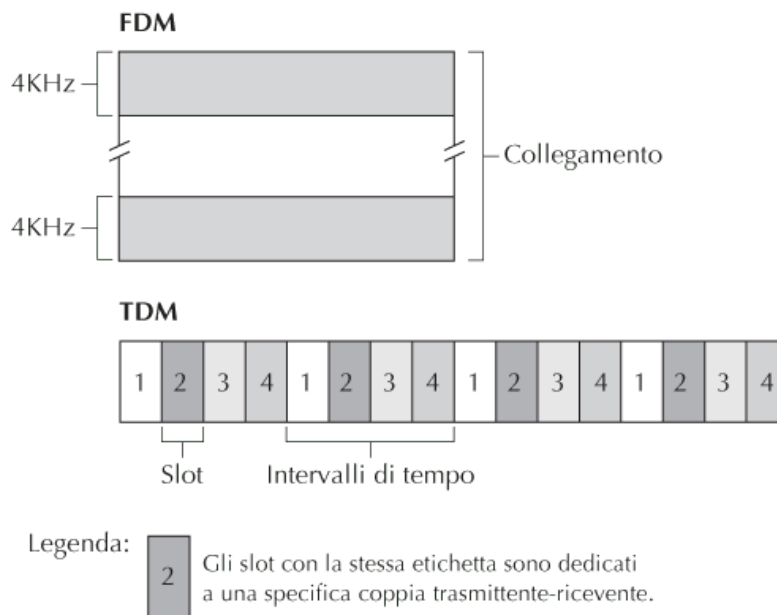


Figura 9: TDM e FDM con 4 nodi

### 1.3.2 Protocolli ad accesso casuale

Un nodo trasmette alla **massima velocità** consentita nel canale,  $R$  bps.

Quando si verifica una collisione, i nodi coinvolti ritrasmettono ripetutamente i loro frame fino a quando raggiungono la destinazione, senza collisioni.

La ritrasmissione non è immediata, ma il nodo **attende un periodo di tempo casuale**

#### Slotted ALOHA

##### Assunzioni:

- Frame grandi  $L$  bit
- Tempo suddiviso in slot da  $\frac{L}{R}$  bit
- I nodi cominciano la trasmissione dei frame solo all'inizio degli slot
- I nodi siano sincronizzati in modo che tutti sappiano quando iniziano gli slot
- Qualora in uno slot si verificasse una collisione, tutti i nodi della rete effettuino la ritrasmissione prima della fine dello slot

**Operazioni:** Quando un nodo riceve un nuovo frame, lo trasmette nello slot successivo

- **Nessuna collisione:** il nodo può inviare nuovo frame nello slot successivo
- **Collisione rilevata:** il nodo ritrasmette frame in ogni slot successivo con probabilità  $p$  fino al successo

##### Vantaggi:

- Slotted ALOHA permette la trasmissione a velocità massima,  $R$  bps
- Altamente decentralizzato: solo gli slot nei nodi devono essere sincronizzati
- Semplice

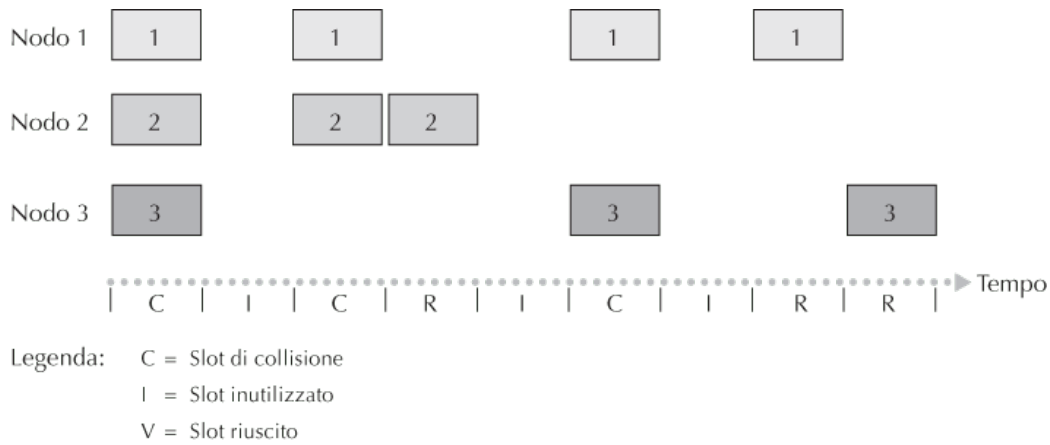


Figura 10: I nodi 1, 2, 3 collidono nel primo slot. Il nodo 2 riesce a trasmettere nel quarto slot, il nodo 1 nell'ottavo e il nodo 3 nel nono.

**Efficienza:** Frazione, sul lungo periodo, di slot con successo quando ci sono molti nodi, ognuno con molti frame da inviare.

Supponiamo ci siano  $N$  nodi con molti frame da inviare: *ognuno trasmette nello slot con probabilità  $p$*

- **Probabilità che il primo nodo abbia successo in slot:**

$$p(1 - p)^{N-1}$$

- **Probabilità che un nodo arbitrario abbia successo in slot:**

$$N \times p(1 - p)^{N-1}$$

- **Efficienza massima:** Trovare  $p'$  tale che massimizzi  $N \times p(1 - p)^{N-1}$

$$- \text{Per molti nodi, } \lim_{N \rightarrow \infty} (N \times p'(1 - p')^{N-1}) = \frac{1}{e} = 0,37$$

**ALOHA puro**