

Università degli Studi di Roma "Tor Vergata"  
Laurea in Informatica

Sistemi Operativi e Reti  
(modulo Reti)  
a.a. 2024/2025

# Livello di collegamento (parte1)

dr. Manuel Fiorelli

[manuel.fiorelli@uniroma2.it](mailto:manuel.fiorelli@uniroma2.it)

<https://art.uniroma2.it/fiorelli>

Basate sulle slide del libro di testo:

[https://gaia.cs.umass.edu/kurose\\_ross/ppt.php](https://gaia.cs.umass.edu/kurose_ross/ppt.php)

Introduction: 1-1

# Livello di collegamento e LAN: obiettivi

- Comprendere i principi alla base dei servizi del livello di collegamento:
  - rilevazione e correzione degli errori
  - condivisione di un canale broadcast: access multiplo
  - indirizzamento a livello di collegamento
  - reti locali: Ethernet, VLAN, reti dei data center
- istanziazione e implementazione di varie tecnologie a livello di collegamento



# Livello di collegamento e LAN: tabella di marcia

- **introduzione**
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- LAN
  - indirizzamento, ARP
  - Ethernet
  - switch
  - VLAN
- canali virtuali: MPLS
- Reti dei data center



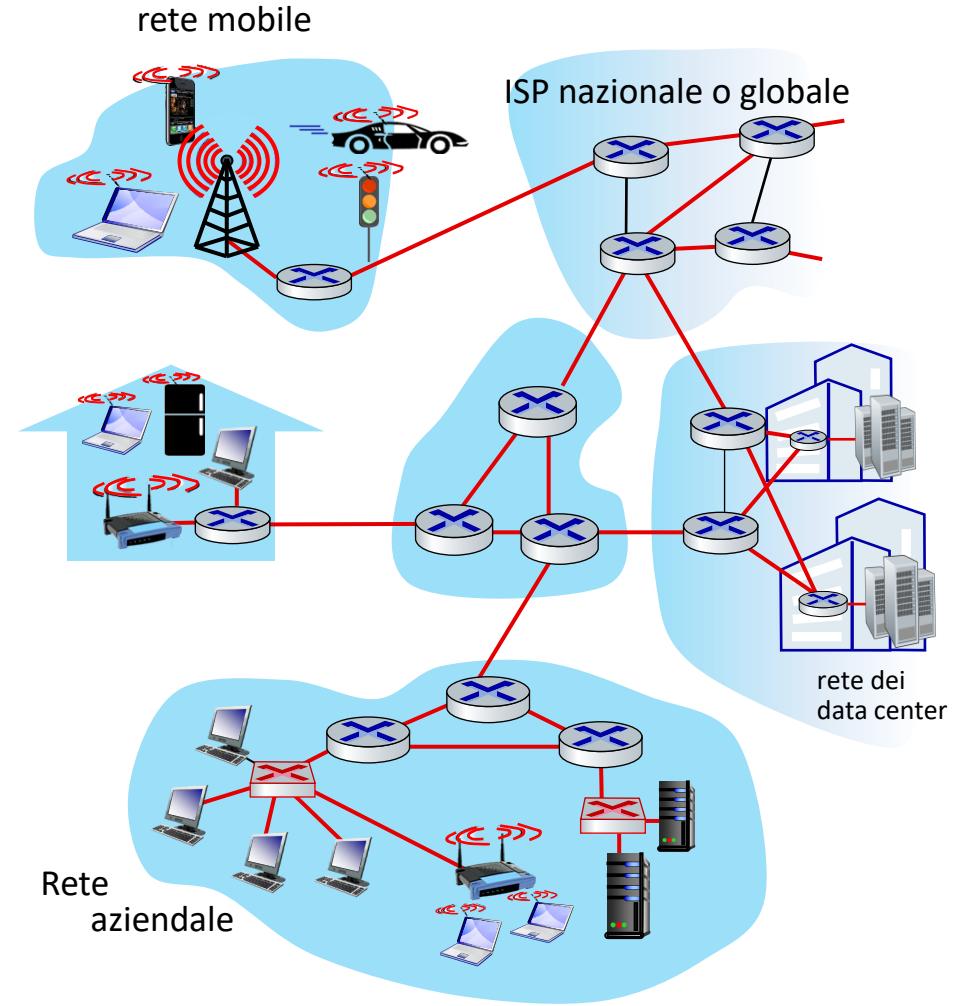
- un giorno nella vita di una richiesta web

# Livello di collegamento: introduzione

terminologia:

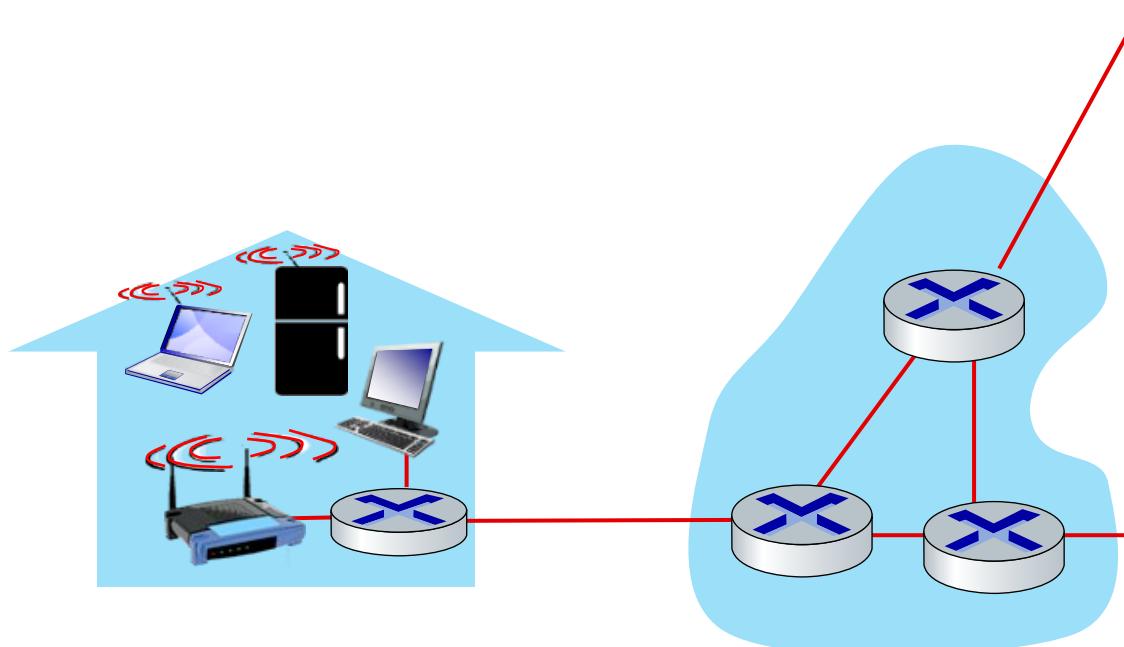
- host, router, switch, etc..: nodi
- canali di comunicazione che collegano nodi **adiacenti** lungo il percorso di comunicazione: **collegamenti (link)**
  - cablati, wireless
  - LAN
- pacchetto di livello 2: **frame**, incapsula datagrammi

*Il livello di collegamento ha la responsabilità di trasferire i datagrammi da un nodo a quello **fisicamente adiacente** lungo un collegamento*

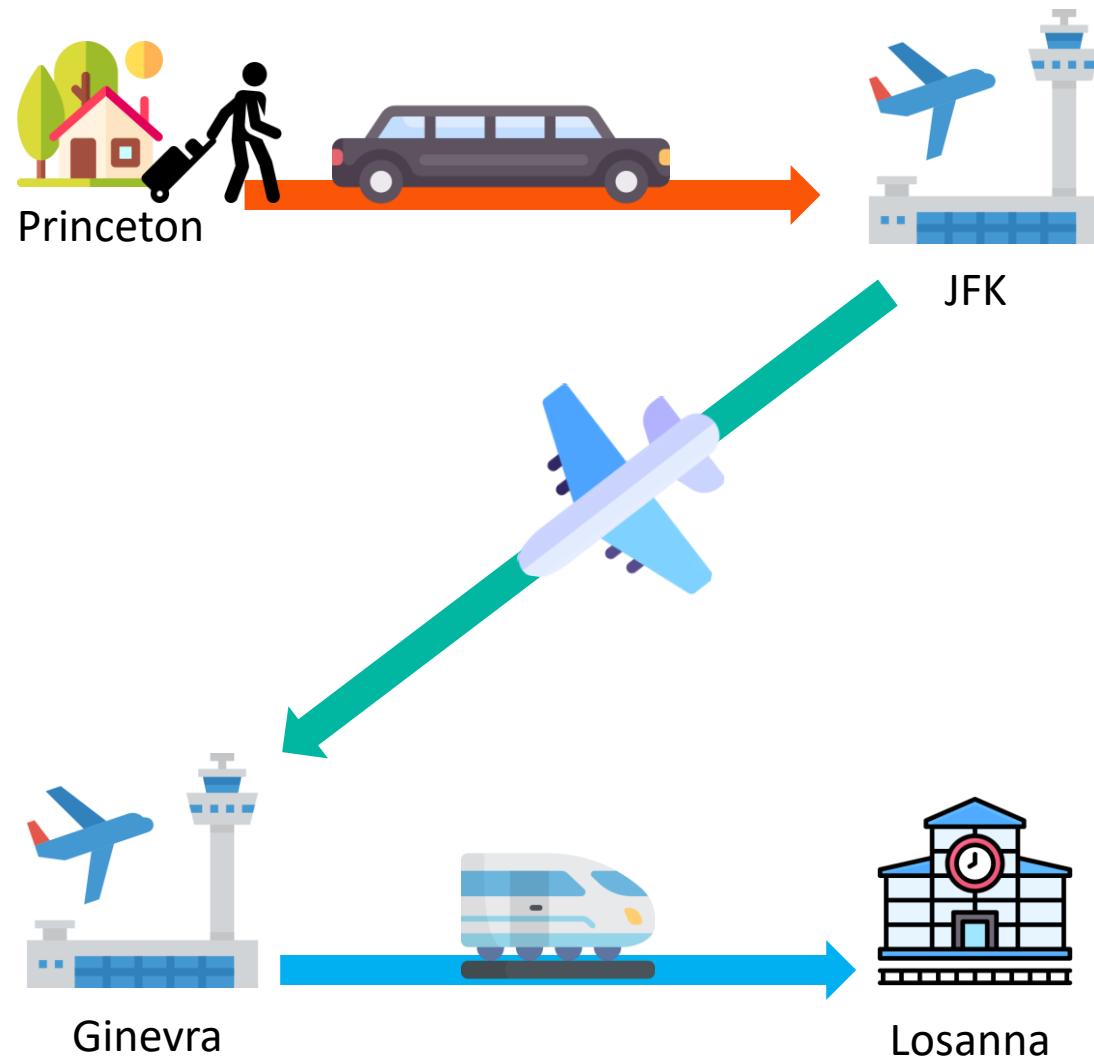


# Livello di collegamento: contesto

- datagramma trasferito da **protocolli di collegamenti differenti** su collegamenti differenti:
  - es., WiFi sul primo collegamento, Ethernet sul collegamento successivo
- ciascun protocollo di collegamento fornisce servizi differenti
  - es. **può o meno** fornire il trasferimento di dati affidabile sul collegamento



# Analogia con i trasporti



## analogia con i trasporti

- viaggio da Princeton a Losanna
  - limo: da Princeton a JFK
  - aereo: da JFK a Ginevra
  - treno: da Geneva a Lausanne
- turista = datagramma
- segmento di trasporto = collegamento
- modalità di trasporto = protocollo a livello di collegamento
- agenzia di viaggi = algoritmo di instradamento

# Livello di collegamento: servizi

- **framing:**

- incapsula i datagrammi in frame, aggiungendo una intestazione e un trailer

- **accesso al collegamento:**

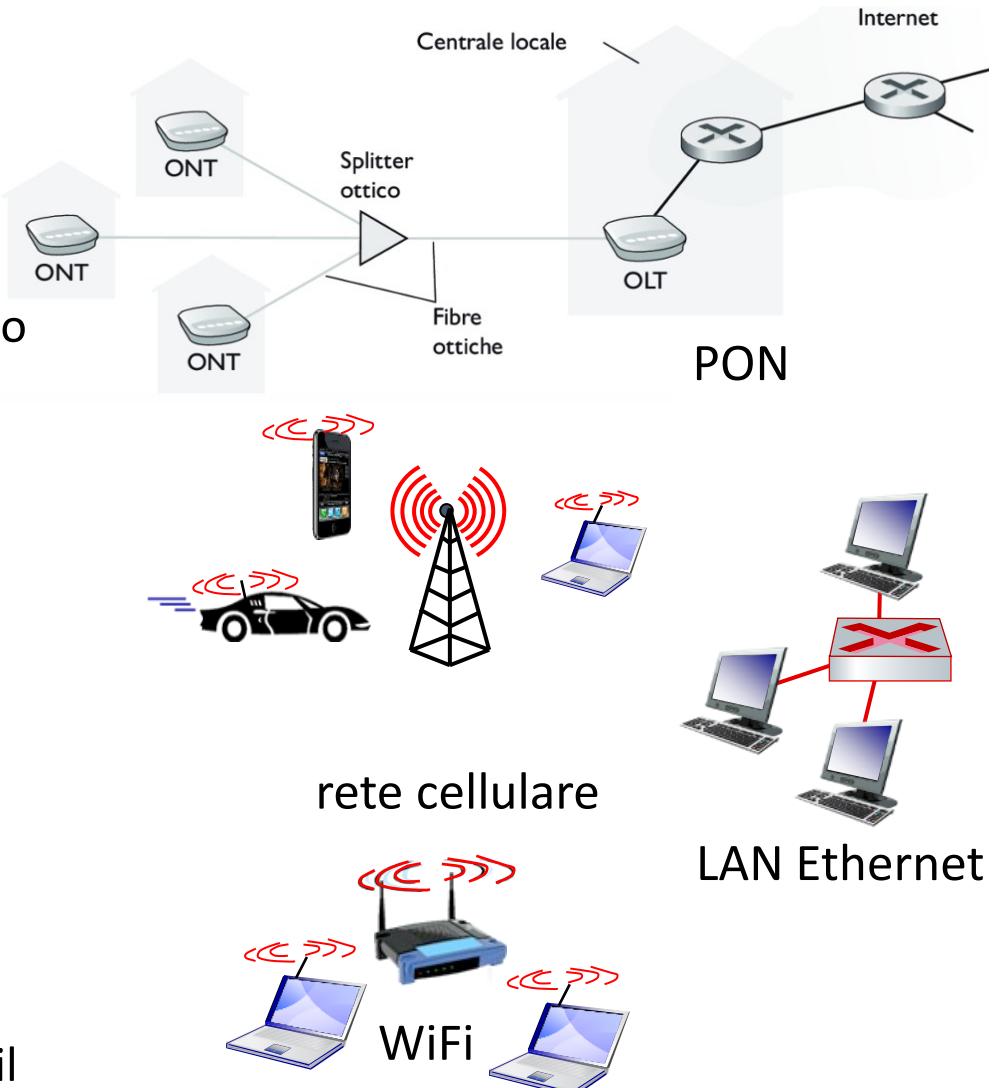
- un protocollo che controlla l'accesso al mezzo trasmittivo (MAC, medium access control) se il mezzo trasmittivo è condiviso (e ci sono più mittenti)
- indirizzi "MAC" nell'intestazione dei frame per identificare la sorgente e la destinazione (diversi dagli indirizzi IP!)

- **half-duplex e full-duplex:**

- con half duplex, i nodi ad entrambi gli estremi del collegamento possono trasmettere, ma non contemporaneamente

- **consegna affidabile tra nodi adiacenti**

- già sappiamo come farlo!
- usato raramente con canali con basso tasso di errore
- collegamenti wireless: tassi di errore elevati
  - correggere l'errore localmente anziché costringere il livello di trasporto o l'applicazione a ritrasmissioni dalla sorgente alla destinazione?



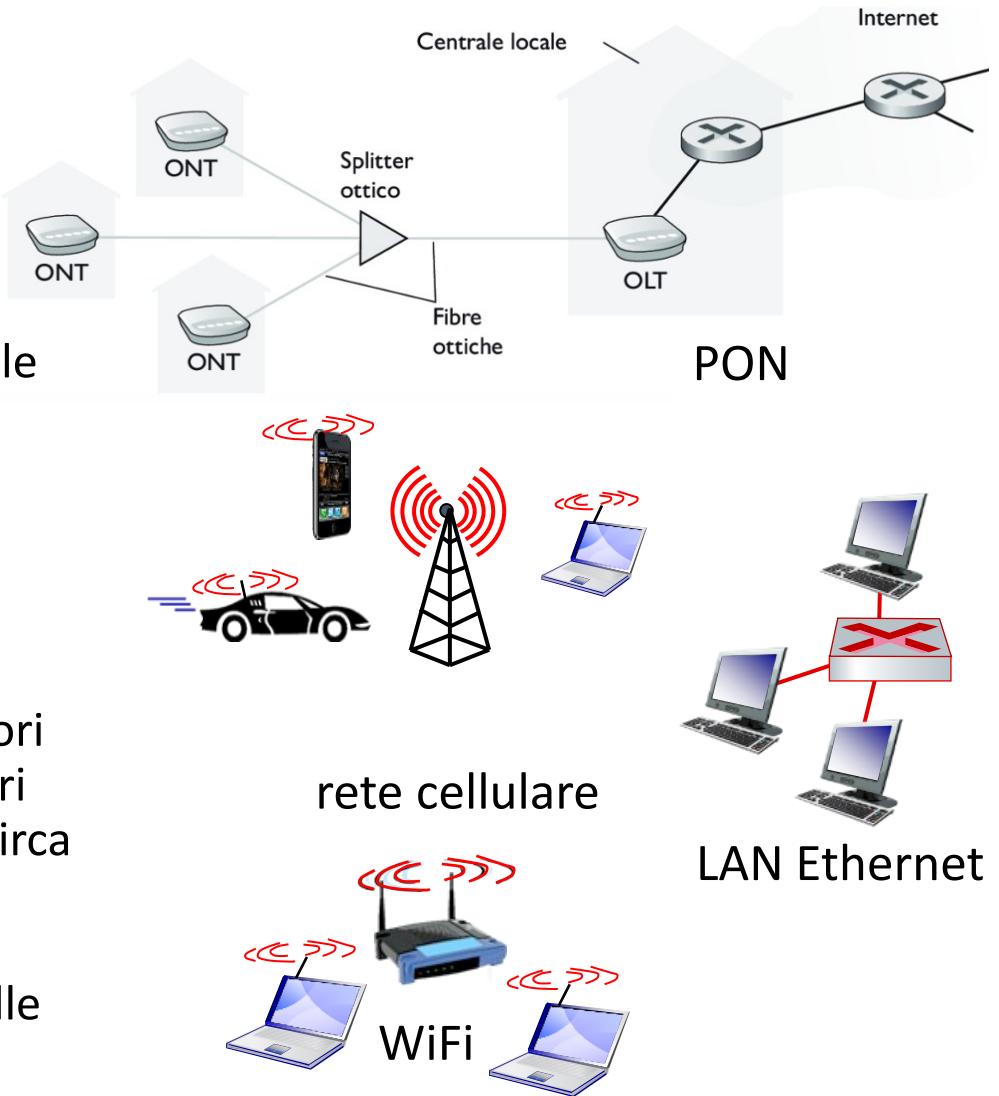
# Livello di collegamento: servizi (continua)

- **controllo di flusso:**

- velocità tra nodi trasmittente e ricevente adiacenti

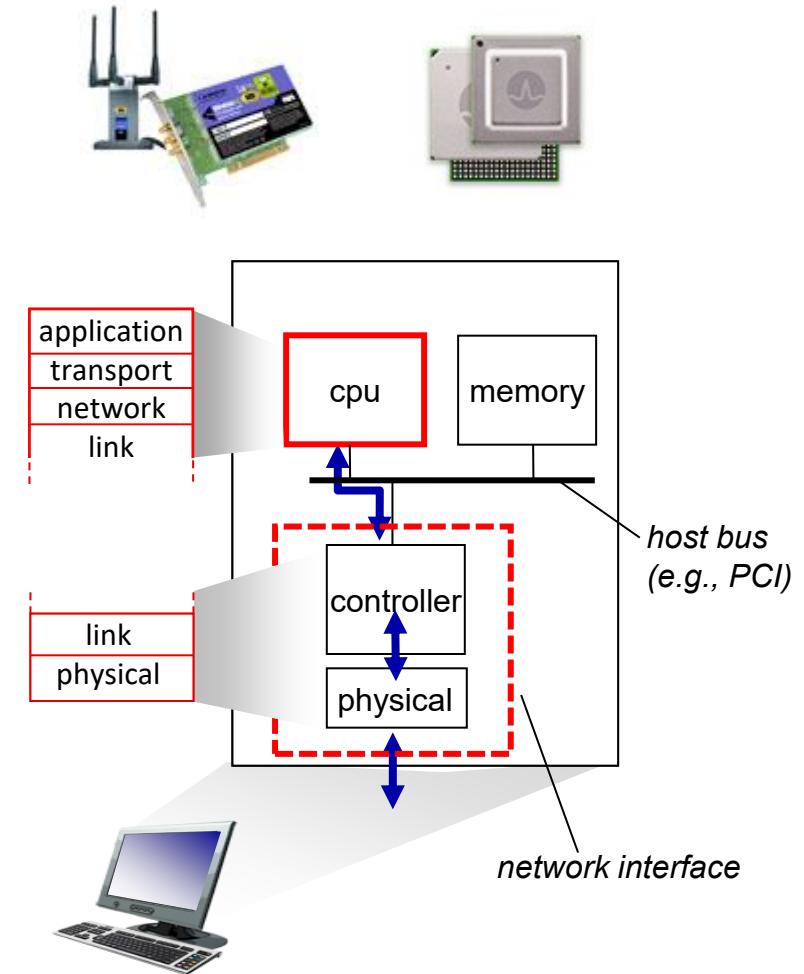
- **rilevazione e correzione degli errori:**

- gli errori sui bit sono causati da attenuazione del segnale e da rumore
- il nodo ricevente rileva gli errori. Due approcci per la correzione:
  - ARQ (*automatic repeat request*): basato su ritrasmissioni
  - *forward error correction* (FEC, correzione degli errori in avanti): il ricevente identifica *e corregge* gli errori sui bit senza ritrasmissioni (evitando un'attesa di circa un RTT, utile per applicazioni in tempo reale o nel caso di RTT molto elevato [si pensi alla comunicazione nello spazio profondo] e al di là delle reti anche nelle applicazioni di storage).

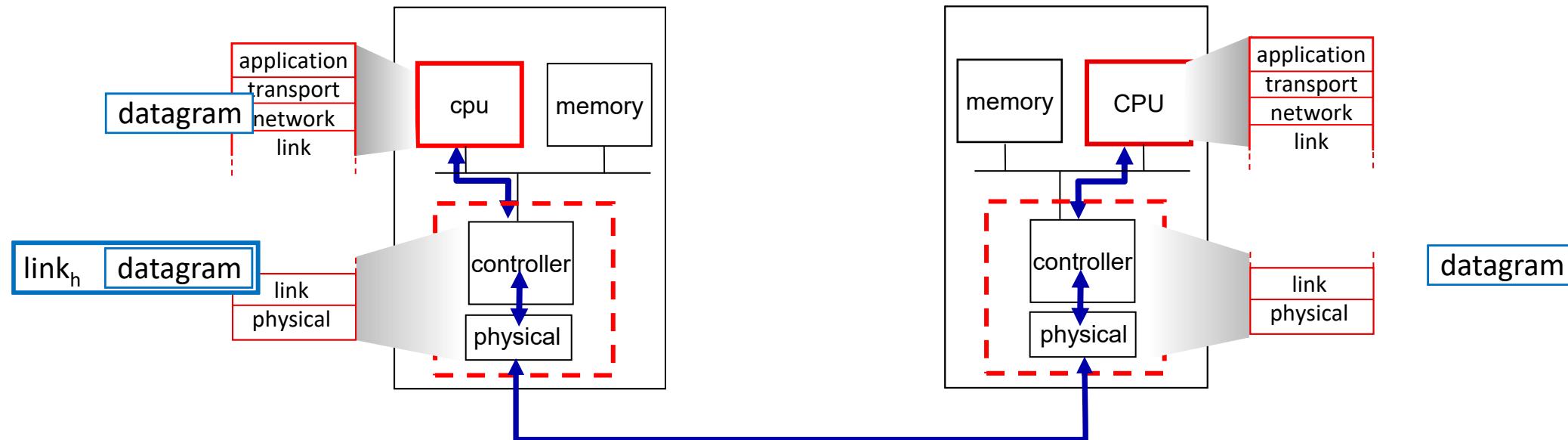


# Implementazione del livello di collegamento negli host

- in ogni singolo host
- il livello di collegamento implementato (principalmente) dall'adattatore di rete (*network adapter*) o scheda di rete (*network interface card*, NIC)
  - implementa il livello di collegamento e quello fisico
  - si collega al *bus* di sistema
- combinazione di hardware, software e firmware



# Adattatore di rete negli host



Lato mittente, il controllore:

- incapsula il datagramma in un frame
- aggiunge bit di controllo degli errori, implementa il trasferimento di dati affidabile, il controllo del flusso, etc.

La CPU esegue la parte software del livello di collegamento, relativa a: interazione con l'adattatore di rete (come dispositivo di IO), assemblaggio delle informazioni di indirizzamento (lato mittente), gestione di condizioni di errore e il passaggio del datagramma fino al livello di rete (lato ricevente).

Lato ricevente, il controllore:

- verifica la presenza di errori e si occupa del trasferimento dati affidabile, del controllo di flusso, etc.
- estrae il datagramma e lo passa al livello superiore (quest'ultimo passato gestito in realtà dal software)

# Livello di collegamento e LAN: tabella di marcia

- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- LAN
  - indirizzamento, ARP
  - Ethernet
  - switch
  - VLAN
- canali virtuali: MPLS
- Reti dei data center

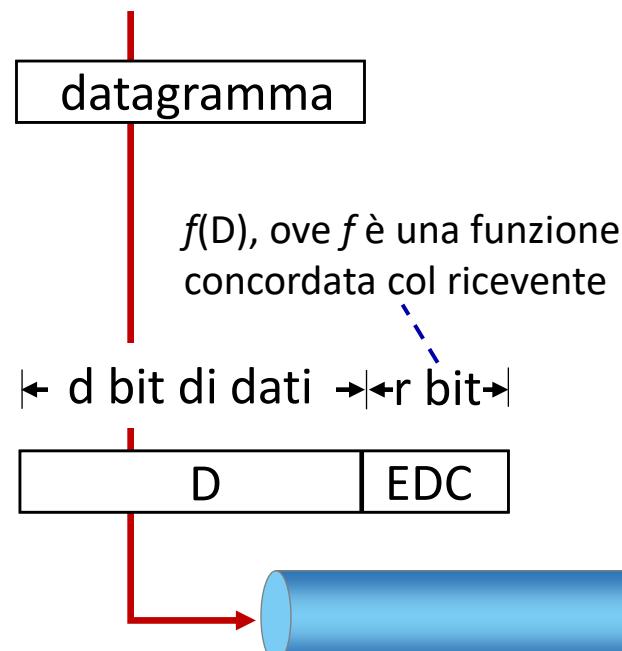


- un giorno nella vita di una richiesta web

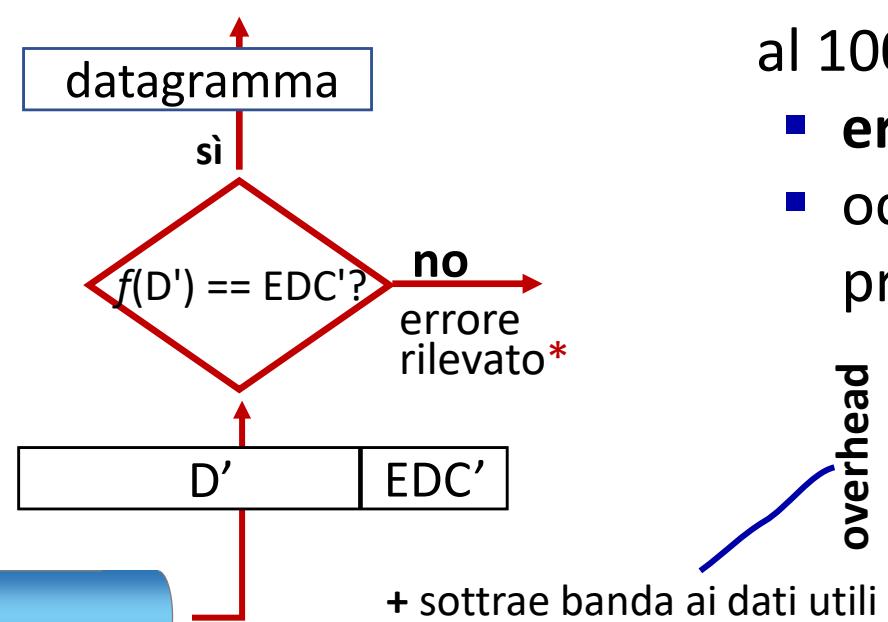
# Rilevazione degli errori

EDC: error detection and correction

D: dati protetti dal controllo d'errore, può includere i campi di intestazione



\*è possibile rilevare errori sia nei dati sia nei bit EDC



*collegamento soggetto a errori sui bit*

+ sottrae banda ai dati utili  
+ ritardo di elaborazione  
(mitigato se implementato in hardware)

Rilevazione non affidabile al 100%!

- **errori non rilevati**
- occorre ridurne la probabilità:
  - più bit di EDC
  - calcoli più complessi

rispetto al livello di trasporto, il livello di collegamento utilizza tecniche più complesse implementate in hardware

# Considerazioni generali (1)

Assumendo errori casuali (non introdotti ad arte) e supponendo che la funzione di calcolo dell'EDC offra una sufficiente capacità di diffusione, possiamo ***in prima approssimazione*** stimare la probabilità condizionata che un errore — una volta verificatosi — non venga rilevato in  $2^{-r}$  (assumendo cioè che i valori di EDC siano distribuiti in maniera uniforme).

Questa è solo una *rule of thumb* che ci aiuta a capire perché è utile incrementare il numero  $r$  di bit EDC. Tuttavia, non descrive le prestazioni di uno specifico EDC soprattutto se si considerano assunzioni più realistiche sulla distribuzione degli errori.

# Considerazioni generali (2)

Per valutare la robustezza dei codici di rilevamento degli errori (EDC), è infatti importante tenere presenti i seguenti aspetti:

- la probabilità di errore su un bit è  $<< 1$  (molto minore di): è assai più probabile che un bit sia ricevuto correttamente piuttosto che essere alterato
- gli errori non sono indipendenti ma spesso si manifestano in raffiche (burst)
- certi pattern di errore potrebbero essere rilevati con certezza mentre altri potrebbero passare sicuramente inosservati

Per questo, nel seguito, analizzeremo il comportamento degli EDC rispetto a diversi pattern di errore.

# Controllo di parità



## Singolo bit di parità (parity bit):

- Rileva un numero *dispari* di errori

0111000110101011	1
------------------	---

←  $d$  bit di dati → |  
bit di parità

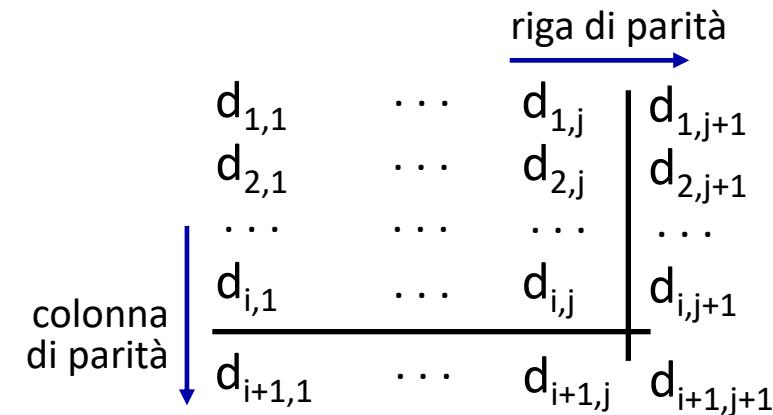
Parità pari/dispari: imposta il bit di parità in modo che ci sia un numero pari/dispari di 1

### Il ricevente:

- calcola la parità dei  $d$  bit ricevuti
- lo confronta con il bit di parità ricevuto – se differente, allora è stato rilevato un errore

## Parità bidimensionale:

- rileva tutte le combinazioni di al più 3 errori
- rilevazione e *correzione* di errori singoli



Senza errori:	1 0 1 0 1   1
	1 1 1 1 0   0
	0 1 1 1 0   1
	0 0 1 0 1   0

Errore su un singolo bit  
rilevato e correggibile:

1 0 1 0 1   1
1 0 1 1 0   0
0 1 1 1 0   1
0 0 1 0 1   0

Erre di parità

# Controllo di parità

Senza errori:  $\begin{array}{r|l} 1 & 0 1 0 1 \\ 1 & 1 1 1 0 \\ 0 & 1 1 1 0 \end{array} \quad \begin{array}{l} 1 \\ 0 \\ 1 \end{array}$

Due errori sulla stessa riga,  
rilevati su colonne differenti

Errore su due bit rilevato ma non correggibile:

1	0	1	0	1	1
1	0	1	1	1	0
0	1	1	1	0	1
					0
0	0	1	0	1	0

Erre<sup>re</sup> di parità  
Erre<sup>re</sup> di parità

Due errori sulla stessa colonna,  
rilevati su righe differenti

Errore su due bit rilevato ma non correggibile:

1	1	0	1	1
1	0	1	1	0
0	1	1	1	0
				1
0	0	1	0	1

Erre<sup>re</sup> di parità  
Erre<sup>re</sup> di parità

Errore su due bit rilevato ma non correggibile:

1	0	1	0	1	1
1	0	1	1	0	0
0	1	1	1	1	1
					0
0	0	1	0	1	0

Erre<sup>re</sup> di parità  
Erre<sup>re</sup> di parità

Errore su due bit rilevato ma non correggibile:

1	0	1	0	1	1
1	1	1	1	1	0
0	0	1	1	0	1
					0
0	0	1	0	1	0

Erre<sup>re</sup> di parità  
Erre<sup>re</sup> di parità

Si noti come i due pattern di errore qui sopra sono differenti, ma ciononostante abbiamo prodotto gli stessi errori di parità! L'ambiguità tra i due casi ci impedisce di correggere l'errore.

# Controllo di parità

Senza errori: 
$$\begin{array}{r|l} 1 & 0 \ 1 \ 0 \ 1 \\ 1 & 1 \ 1 \ 1 \ 0 \\ 0 & 1 \ 1 \ 1 \ 0 \\ \hline 0 & 0 \ 1 \ 0 \ 1 \end{array}$$

Errore su tre bit rilevato ma non correggibile:

$$\begin{array}{r|l} 1 & 1 \ 1 \ 0 \ 1 \ 1 \\ 1 & 0 \ 1 \ 1 \ 1 \ 0 \\ 0 & 1 \ 1 \ 1 \ 0 \ 1 \\ \hline 0 & 0 \ 1 \ 0 \ 1 \ 0 \end{array}$$

Errore di parità

Errore su tre bit rilevato ma non correggibile:

$$\begin{array}{r|l} 1 & 1 \ 1 \ 0 \ 1 \ 1 \\ 1 & 1 \ 1 \ 1 \ 0 \ 0 \\ 0 & 0 \ 1 \ 1 \ 1 \ 1 \\ \hline 0 & 0 \ 1 \ 0 \ 1 \ 0 \end{array}$$

Errore di parità

Errore su tre bit rilevato ma non correggibile:

$$\begin{array}{r|l} 1 & 0 \ 0 \ 0 \ 1 \ 1 \\ 1 & 0 \ 1 \ 1 \ 0 \ 0 \\ 0 & 1 \ 1 \ 1 \ 1 \ 1 \\ \hline 0 & 0 \ 1 \ 0 \ 1 \ 0 \end{array}$$

Errore di parità

Errore di parità

Errore di parità

# Controllo di parità

Senza errori: 1 0 1 0 1 | 1  
1 1 1 1 0 | 0  
0 1 1 1 0 | 1  
—————  
0 0 1 0 1 | 0

Errore su quattro bit  
**non rilevato**

1	0	1	0	1		1					
1	0	1	1	1		0					
0	0	1	1	1		1					
					0	0	1	0	1		0



Ma non significa che non si possano rilevare *alcuni* errori su quattro bit

Errore su quattro bit  
**rilevato ma non**  
**correggibile:**

0	0	1	0	1		1	→					
1	0	1	1	0		0	→					
0	1	0	1	0		1	→					
					0	0	1	1	0		0	→

Errori di parità

# Controllo di parità

- adatto quando entrambe le seguenti condizioni sono vere:

- la probabilità di errori nei bit è bassa
- gli errori sono indipendenti

Sotto queste ipotesi, la probabilità di errori multipli è molto bassa: pertanto è improbabile che si verifichi un numero pari di errori non rilevati dal bit di parità.

- nella realtà, però, gli errori tendono a verificarsi in *burst*

- la probabilità che errori a burst non siano rilevati da un singolo bit di parità può avvicinarsi al 50%

dati inviati

0	0	1	1	0	1	0	1	0	0
---	---	---	---	---	---	---	---	---	---

dati ricevuti

0	0	1	0	0	0	1	1	0	1
---	---	---	---	---	---	---	---	---	---

lunghezza errore a burst (nell'esempio 7 bit)

# Checksum Internet (ripasso, si veda sezione 3.3)

**Obiettivo:** rilevare gli "errori" (*bit alternati*) nel segmento trasmesso  
**mittente:**

- tratta il contenuto del segmento come una sequenza di interi a 16 bit (inclusi i campi dell'intestazione UDP e gli indirizzi IP)
- **checksum:** complemento a 1 della somma (in complemento a 1) della sequenza di interi a 16 bit
- pone il valore del checksum nel campo checksum del segmento UDP

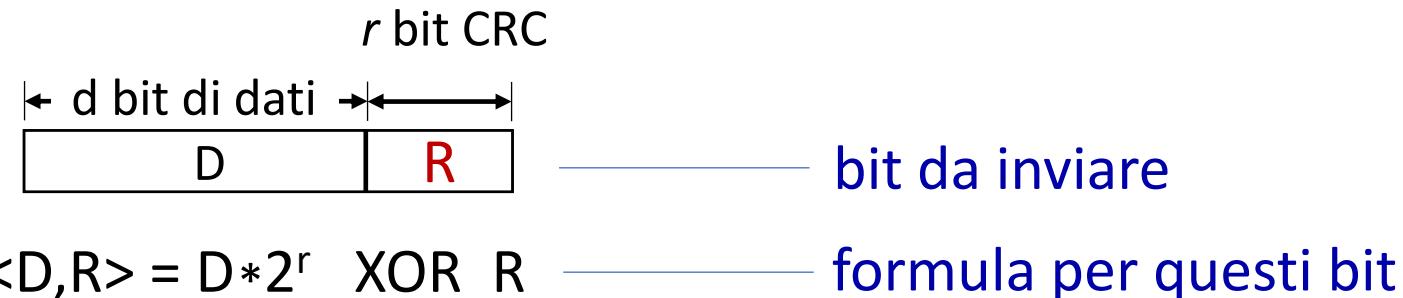
**ricevente:**

- calcola la somma in complemento a 1 allo stesso modo del mittente, includendo però il checksum ricevuto
- il risultato è costituito da tutti bit 1 (-0 nell'aritmetica del complemento a 1)?
  - Sì - nessun errore rilevato
  - No - errore rilevato
- oppure, si esegue il complemento a 1 finale: si è calcolato il checksum di tutti i dati ricevuti (incluso il checksum stesso) e si verifica che sia formata da soli 0

# Codici di controllo a ridondanza ciclica (Cyclic Redundancy Check, CRC)

- codifica di rilevamento degli errori più potente
- D: *dati* da trasmettere (d bit)
- G: sequenza di  $(r + 1)$  bit concordata, detta *generatore* (definito nello standard CRC)

il bit più significativo deve essere 1; nei generatori di uso pratico, lo è anche quello meno significativo



**mittente:** calcola  $r$  bit CRC, R, tali che  $\langle D, R \rangle$  si *divisibile esattamente* da G (mod 2)

- il ricevente conosce G, divide  $\langle D, R \rangle$  per G. Se il resto è diverso da zero: errore rilevato!
- può rilevare tutti gli errori a burst di lunghezza inferiore a  $r+1$  bit (ovvero, errori di non più di  $r$  bit consecutivi)
- la frazione dei burst più lunghi che può rilevare è approssimativamente  $1 - 2^{-r}$
- largamente usato in pratica (Ethernet, 802.11 WiFi)

# Codici di controllo a ridondanza ciclica (Cyclic Redundancy Check, CRC)

tutti i calcoli di CRC sono eseguiti in aritmetica modulo 2 senza riporti nelle addizioni e prestiti nelle sottrazioni

- addizione e sottrazione sono la stessa operazione, corrispondente all'or esclusivo (*exclusive or*, XOR) bit a bit

$$\begin{array}{r} 1011 + \\ \underline{1101 =} \\ 0110 \end{array} \quad \begin{array}{r} 1011 - \\ \underline{1101 =} \\ 0110 \end{array} \quad 1011 \text{ XOR } 1101 = 1011 \oplus 1101 = 0110$$

- la moltiplicazione e la divisione sono calcolate come al solito, usando queste definizioni di addizione e sottrazione

$$\begin{array}{r} 1011 \times \\ 101 = \\ \hline 1011 + \\ 0 + \\ \hline 1011 = \\ 100111 \end{array}$$

Questa "strana" aritmetica corrisponde al vedere le sequenze di bit come i coefficienti (modulo 2) di polinomi

# Codici di controllo a ridondanza ciclica (Cyclic Redundancy Check, CRC)

assumendo che il primo bit sia 1, allora il *grado* del polinomio è uguale al numero di bit - 1

$$\begin{array}{rcl} 1011 \times & \xrightarrow{\hspace{2cm}} & x^3 + x + 1 \\ \underline{101 =} & \xrightarrow{\hspace{2cm}} & x^2 + 1 \\ 1011 + & \xrightarrow{\hspace{2cm}} & x^3 + x + 1 \\ 0 + & & \\ \underline{1011 =} & \xrightarrow{\hspace{2cm}} & (x^3+x+1) \cdot x^2 = x^5 + x^3 + x^2 \\ 100111 & \xrightarrow{\hspace{2cm}} & x^5 + x^2 + x + 1 \end{array}$$

modulo 2

$\left. \begin{aligned} &+ = (x^5 + x^3 + x^2) + (x^3 + x + 1) \\ &= x^5 + (1 + 1)x^3 + x^2 + x + 1 \\ &= x^5 + (1 + 1)x^3 + x^2 + x + 1 \\ &= x^5 + x^2 + x + 1 \end{aligned} \right\}$

# Codici di controllo a ridondanza ciclica: esempio

Il mittente vuole calcolare R tale che:

$$D \cdot 2^r \text{ XOR } R = nG$$

... o equivalentemente (XOR R in entrambi i lati):

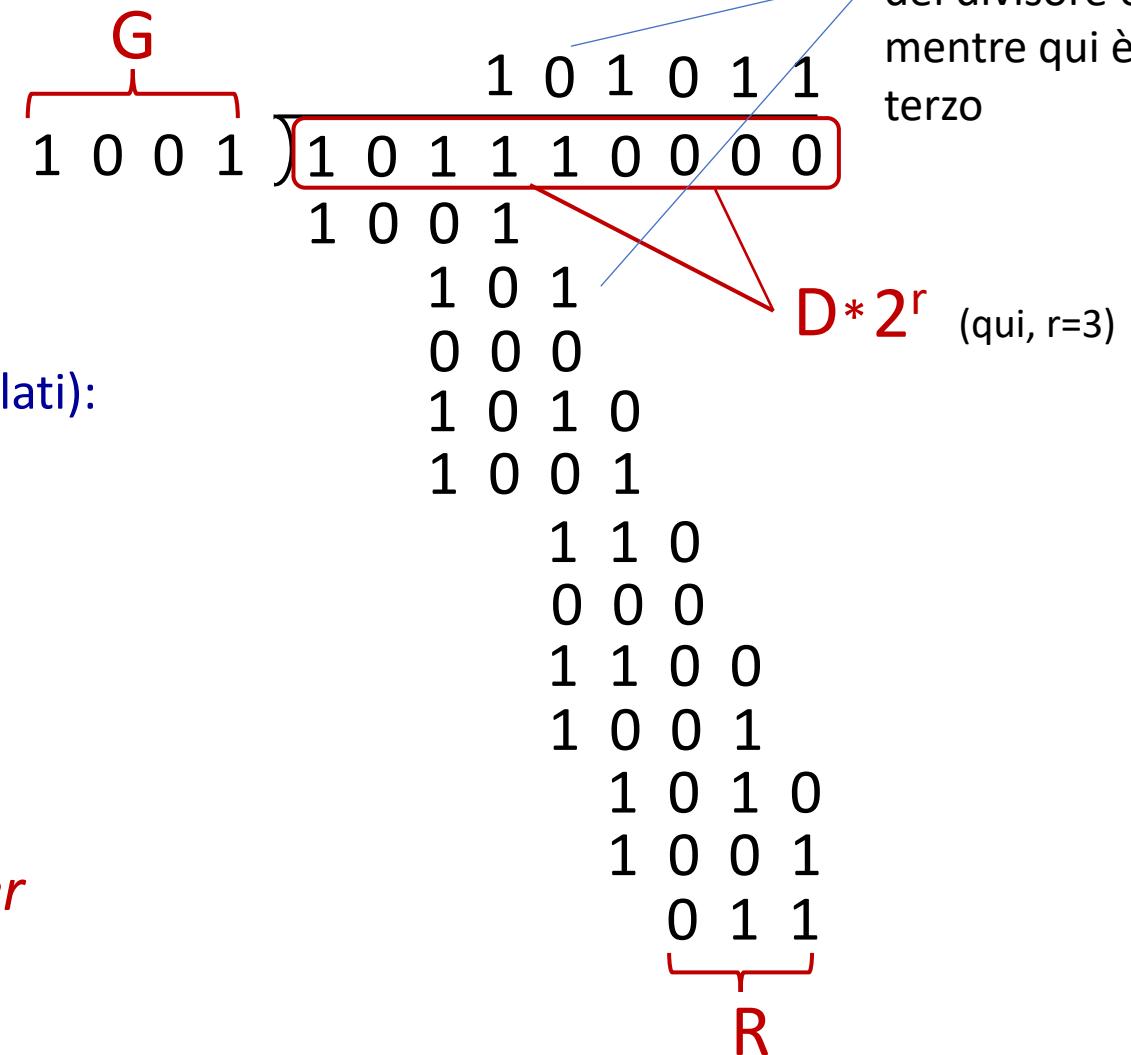
$$D \cdot 2^r = nG \text{ XOR } R$$

... che ci dice che:

se dividiamo  $D \cdot 2^r$  per G il resto è precisamente R

$$R = \text{resto} \left[ \frac{D \cdot 2^r}{G} \right]$$

*algoritmo per calcolare R*



Il bit 1 più alto del divisore è il 4, mentre qui è il terzo

\* Check out the online interactive exercises for more examples: [http://gaia.cs.umass.edu/kurose\\_ross/interactive/](http://gaia.cs.umass.edu/kurose_ross/interactive/)

# CRC: polinomio di errore e scelta del generatore

il mittente invia  $T = D \cdot 2^r \text{ XOR } R$

il ricevente riceve  $T' = T \text{ XOR } E = T + E$  (usando l'addizione CRC)

dove  $E$  è una sequenza di bit (ovvero un polinomio), i cui bit a 1 indicano dove si è verificato un errore.

Siccome  $T$  è divisibile per  $G$ ,  $T'$  sarà divisibile per  $G$  se e solo se  $E$  è divisibile per  $G$ .

Quindi, *G deve essere scelto in modo tale che NON divida i polinomi di errore.*

In pratica, si cerca di definire  $G$  in modo che rilevi diversi tipi di errore.

# CRC: polinomio di errore e scelta del generatore

Se  $G$  ha un numero pari di bit a 1, allora è in grado di rilevare qualsiasi numero dispari di errori.  
Perché?

- $E(x)$  il polinomio che rappresenta l'errore (con un numero **dispari** di bit a 1),
- $G(x)$  il polinomio generatore (con un numero **pari** di bit a 1).

Supponiamo per assurdo che l'errore passi inosservato, cioè che  
 $E(x) = Q(x) \cdot G(x)$  per qualche polinomio  $Q(x)$ .

Valutiamo entrambi i lati in  $x = 1$

- poiché  $E(x)$  contiene un numero dispari di bit a 1, si ha  $E(1) = 1$
- poiché  $G(x)$  contiene un numero pari di bit a 1, si ha  $G(1) = 0$ .

Ne segue

$$E(1) = Q(1) \cdot G(1) \Rightarrow 1 = Q(1) \cdot 0, \text{ che è una contraddizione.}$$

Quindi  $E(x)$  **non** è divisibile per  $G(x)$  e pertanto l'errore sarà sicuramente rilevato

Nota: il caso più semplice è  $G(x) = x + 1$  (in bit "1"), che corrisponde al classico controllo di parità.

# CRC: polinomio di errore e scelta del generatore

Se  $G$  ha almeno due bit a 1, esso è in grado di rilevare qualunque errore singolo.

Perché?

Un errore singolo è espresso dal polinomio  $x^k$  (cioè  $1 \overbrace{0000 \dots 0}^k$ ) che è divisibile solo dai polinomi  $x^i$  per  $i \leq k$ , essendo  $x^k = x^{k-i} \cdot x^i$ .

Università degli Studi di Roma "Tor Vergata"  
Laurea in Informatica

Sistemi Operativi e Reti  
(modulo Reti)  
a.a. 2024/2025

# Livello di collegamento (parte2)

dr. Manuel Fiorelli

[manuel.fiorelli@uniroma2.it](mailto:manuel.fiorelli@uniroma2.it)

<https://art.uniroma2.it/fiorelli>

Basate sulle slide del libro di testo:

[https://gaia.cs.umass.edu/kurose\\_ross/ppt.php](https://gaia.cs.umass.edu/kurose_ross/ppt.php)

Introduction: 1-28

# Livello di collegamento e LAN: tabella di marcia

- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- LAN
  - indirizzamento, ARP
  - Ethernet
  - switch
  - VLAN
- canali virtuali: MPLS
- Reti dei data center



- un giorno nella vita di una richiesta web

# Collegamenti e protocolli di accesso multiplo

Due tipi di collegamenti:

- punto a punto (*point-to-point*): un trasmettente a un'estremità del collegamento e un unico ricevente all'altra estremità
  - collegamento punto a punto tra host e switch Ethernet
  - protocollo PPP per accesso dial-up
- **broadcast**: un canale broadcast **condiviso** tra più nodi trasmittenti e riceventi, ciascun frame viene ricevuto da tutti i nodi
  - Ethernet "vecchia scuola" (cavo condiviso)
  - 802.11 wireless LAN, 4G/5G, satellite



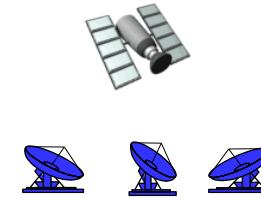
cavo condiviso (es.,  
Ethernet cablata)



spettro radio  
condiviso: 4G/5G



spettro radio  
condiviso: WiFi



spettro radio  
condiviso : satellite



umani a una festa  
(aria condivisa, acustico)

# Protocolli di accesso multiplo

- singolo canale broadcast condiviso
- due o più trasmissioni simultanee dai nodi: interferenza
  - *collisione* se un nodo riceve due o più segnali nello stesso istante

## Protocollo di accesso multiplo

- algoritmo distribuito che determina come i nodi condividono il canale, determina quando i nodi possono trasmettere
- la comunicazione sulla condivisione del canale deve utilizzare il canale stesso!
  - nessun canale fuori banda per il coordinamento

# Un protocollo di accesso multiplo ideale

*dato:* un canale ad accesso multiplo (*multiple access channel*, MAC) con velocità di  $R$  bps

*desiderata:*

1. quando un solo nodo vuole trasmettere, può inviare a velocità  $R$ .
2. quando  $M$  nodi vogliono trasmettere, ciascun può inviare a una *velocità media*  $R/M$ .
3. totalmente decentralizzato:
  - nessun nodo speciale che coordina le trasmissioni (il cui fallimento potrebbe bloccare il sistema);
  - nessuna sincronizzazione degli orologi, slot temporali, etc.
4. semplice.

# Protocolli di accesso multiplo: tassonomia

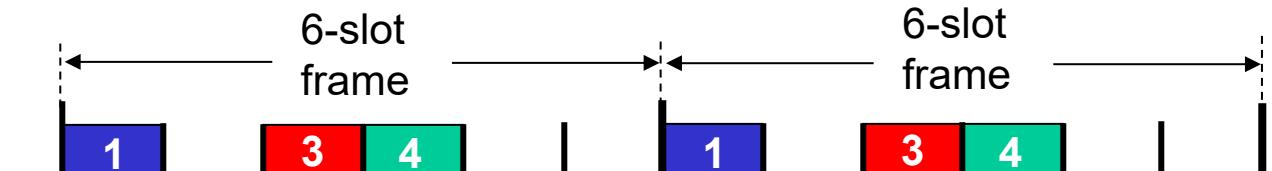
Tre ampie classi:

- **a suddivisione del canale (*channel partitioning*)**
  - divide il canale in "pezzi" più piccoli (slot temporali, bande di frequenza, codici)
  - assegna un pezzo a un nodo per uso esclusivo
- **ad accesso casuale (*random access*)**
  - canale non diviso, permette le collisioni
  - recupera ("recover") dalle collisioni (attraverso ritrasmissioni)
- **a rotazione ("*taking turns*")**
  - i nodi si avvicendano a turno, ma i nodi con una quantità maggiore di materiale da inviare possono fare turni più lunghi

# Protocolli a suddivisione del canale: TDMA

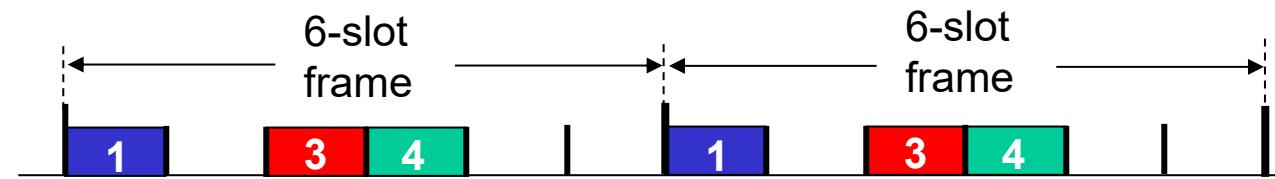
TDMA: time division multiple access (accesso multiplo a divisione di tempo)

- accesso al canale in “intervalli di tempo” (*time frame*)
- ciascun intervallo è ulteriormente suddiviso in N slot temporali (*time slot*), ciascun assegnato a uno degli N nodi
- la durata di uno slot temporale è in genere tale da consentire la trasmissione di un pacchetto a livello di collegamento
- gli slot inutilizzati rimangono inutilizzati (*idle*)
- esempio: LAN con 6 nodi, 1,3,4 hanno pacchetti da inviare, gli slot 2,5,6 sono inattivi



# Protocolli a suddivisione del canale: TDMA

TDMA: time division multiple access (accesso multiplo a divisione di tempo)

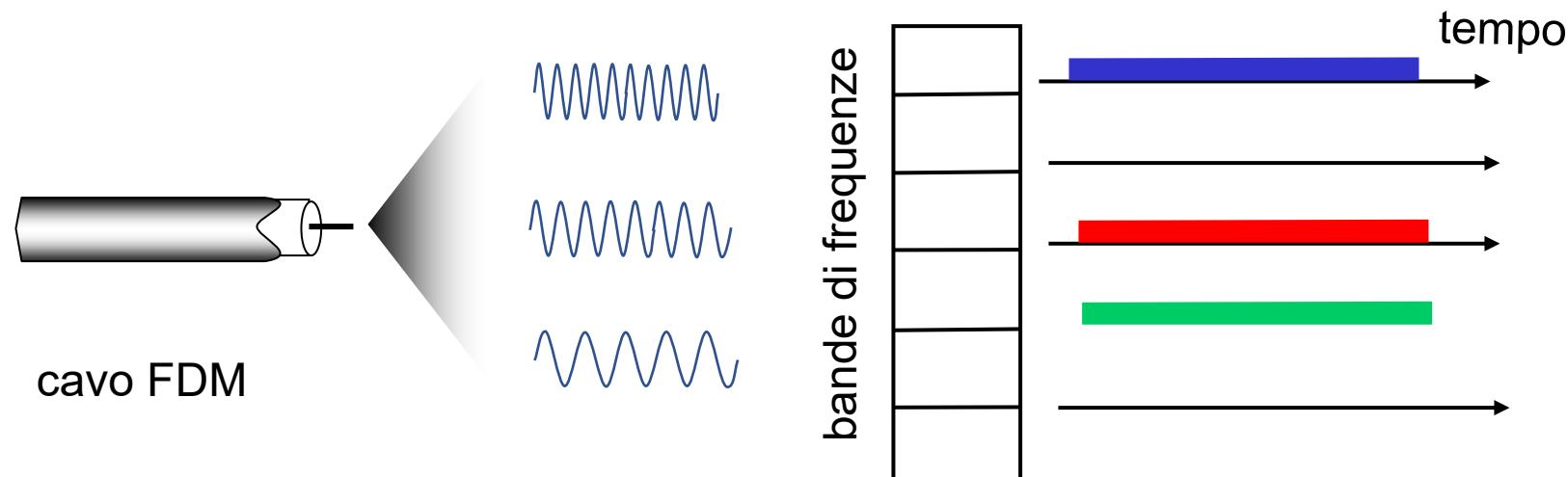


- un nodo che ha dati da trasmettere deve attendere il proprio turno (cioè lo slot assegnatogli)
- nel proprio turno, un nodo trasmette a  $R$  bps, ma potendo farlo solo in  $1/N$  della durata dell'intervallo temporale, la sua velocità media è  $R/N$ ... a prescindere dal fatto che ci siano altri nodi che vogliono trasmettere sul canale

# Protocolli a suddivisione del canale: FDMA

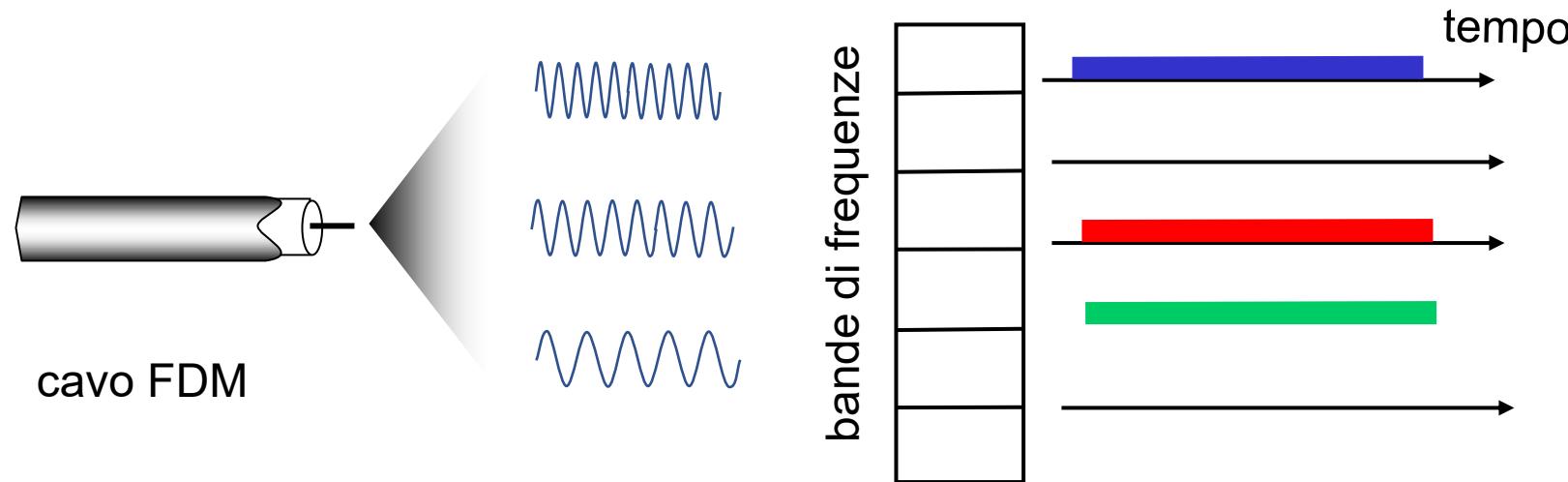
FDMA: frequency division multiple access (accesso multiplo a divisione di frequenza)

- lo spettro del canale diviso in bande di frequenza
- a ciascun nodo viene assegnata una banda di frequenze fisse
- il tempo di trasmissione non utilizzato nelle bande di frequenza resta inutilizzato
- esempio: LAN con 6 nodi, 1,3,4 hanno pacchetti da inviare, le bande di frequenza 2,5,6 sono inutilizzate (*idle*)



# Protocolli a suddivisione del canale: FDMA

FDMA: frequency division multiple access (accesso multiplo a divisione di frequenza)

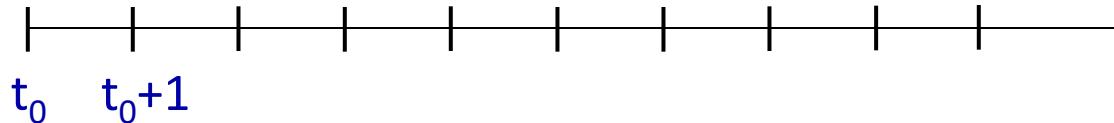


- un nodo può trasmettere nella propria banda di frequenze appena ha dati da inviare, senza dover attendere turni
- Trasmettendo alla velocità massima consentita dalla sua banda di frequenze ridotta, la velocità è R/N a prescindere che altri vogliano trasmettere

# Protocolli ad accesso casuale

- quando un nodo ha un pacchetto da inviare
  - trasmette alla massima velocità consentita dal canale, cioè  $R$  bps
  - nessun coordinamento *a priori* tra i nodi
- due o più nodi stanno trasmettendo nello stesso momento: “collisione”
- un **protocollo ad accesso casuale** specifica:
  - come rilevare le collisioni
  - come recuperare dalle collisioni (es. attraverso ritrasmissioni con ritardo casuale)
- Esempi di protocolli ad accesso casuale:
  - ALOHA, slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA

# Slotted ALOHA



assunzioni:

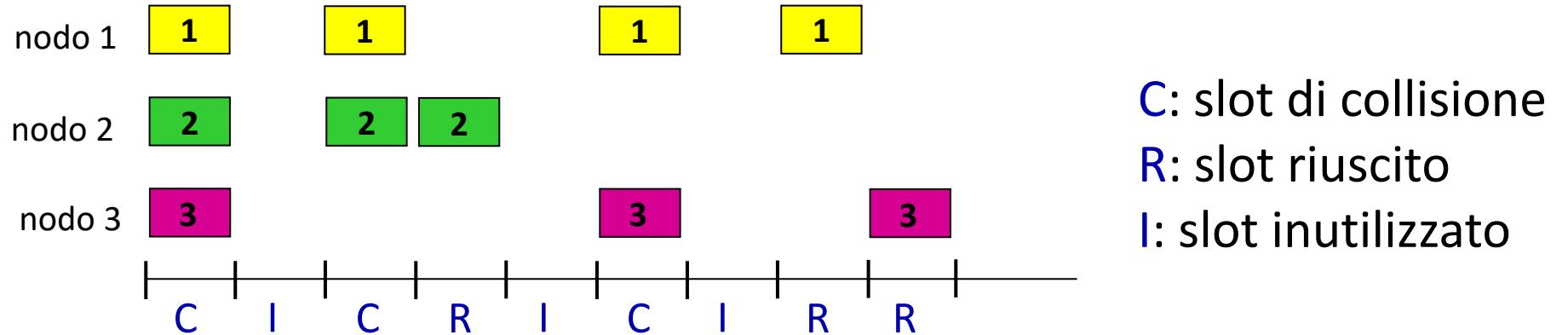
- tutti i frame hanno la stessa dimensione
- tempo suddiviso in slot temporali uguali (equivalenti al tempo per trasmettere in frame)
- i nodi cominciano la trasmissione soltanto all'inizio degli slot
- i nodi sono sincronizzati
- se 2 o più nodi trasmettono nello stesso slot, tutti i nodi rilevano la collisione prima del termine dello slot (es., attraverso la mancata ricezione di un ACK)

operazioni:

- quando un nodo ha un nuovo frame da spedire, lo trasmette all'inizio dello slot successivo
  - *se non si verifica una collisione:* il nodo può inviare un nuovo frame nello slot successivo
  - *se si verifica una collisione:* il nodo ritrasmette il frame nello slot successivo con probabilità  $p$  finché non ha successo

randomizzazione – perché?

# Slotted ALOHA



## Pro:

- un singolo nodo attivo può trasmettere continuamente alla massima velocità del canale
- altamente decentralizzato: solo gli slot nei nodi devono essere sincronizzati
- semplice

## Contro:

- collisioni, spreco di slot
- slot inutilizzati (a causa della politica di trasmissione probabilistica)
- i nodi potrebbero essere in grado di rilevare la collisione in meno del tempo necessario per trasmettere il pacchetto
- sincronizzazione degli orologi

# Slotted ALOHA: efficienza

**efficienza:** frazione a lungo termine di slot riusciti (molti nodi, tutti con molti frame da inviare)

- *si supponga:*  $N$  nodi con molti frame da inviare, ciascuno trasmette nello slot con probabilità  $p$ : il numero di nodi che trasmettono in uno slot è una variabile aleatoria binomiale  $B(N,p)$

- probabilità che un dato nodo ha successo in uno slot =  $p(1-p)^{N-1}$
- probabilità che un nodo qualunque abbia successo =  $Np(1-p)^{N-1}$
- efficienza massima: trovare  $p^*$  che massimizza  $Np(1-p)^{N-1}$
- per molti nodi, calcolare il limite di  $Np^*(1-p^*)^{N-1}$  per  $N$  che tende all'infinito, si ottiene:

$$\text{efficienza massima} = 1/e = .37$$

- *al massimo:* canale usato per trasmissione utile solo per il 37% del tempo!



# Slotted ALOHA: efficienza

Data la formula dell'efficienza

$$\text{efficienza}(p) = Np(1 - p)^{N-1}$$

Ne calcolo la derivata parziale rispetto a  $p$ , e ne studio il segno

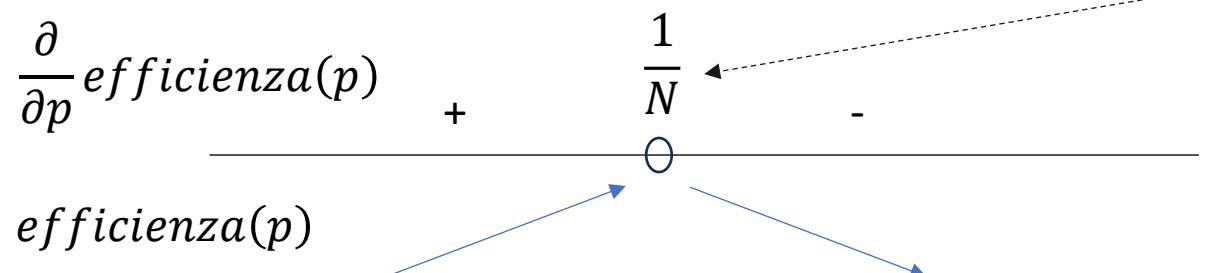
$$\begin{aligned}\frac{\partial}{\partial p} \text{efficienza}(p) &= \frac{\partial}{\partial p} Np(1 - p)^{N-1} = N((1 - p)^{N-1} - p(N - 1)(1 - p)^{N-2}) \\ &= N(1 - p)^{N-2}((1 - p) - p(N - 1)) \geq 0\end{aligned}$$

I primi due fattori sono sempre positivi (assumendo  $p \in (0,1)$ ), pertanto mi concentro sul terzo fattore.

$$(1 - p) - p(N - 1) \geq 0$$

$$1 - p - pN + p \geq 0$$

$$p \leq \frac{1}{N}$$



Massimo della funzione *efficienza*

$$p^* = \frac{1}{N}$$

$$\text{eff}(p^*) = \text{eff}\left(\frac{1}{N}\right) = \left(1 - \frac{1}{N}\right)^{N-1}$$

# Slotted ALOHA: efficienza

Efficienza massima per un *gran numero di nodi*

$$\lim_{N \rightarrow \infty} \text{efficienza}(p^*) = \lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right)^{N-1} = \lim_{N \rightarrow \infty} \frac{\left(1 - \frac{1}{N}\right)^N}{1 - \frac{1}{N}} = \frac{1}{e} = 37\%$$

visto che:

- $\lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right)^N = \lim_{N \rightarrow \infty} \left(1 + \frac{-1}{N}\right)^N = e^{-1} = \frac{1}{e}$

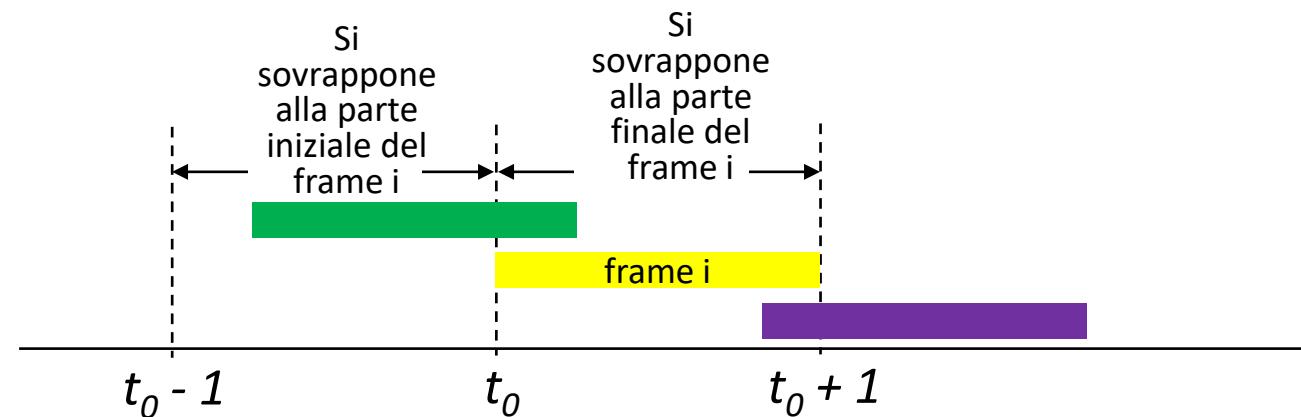
Dato il limite notevole  $\lim_{x \rightarrow \pm\infty} \left(1 + \frac{a}{x}\right)^{bx} = e^{ab}$  [https://it.wikipedia.org/wiki/Limite\\_notevole](https://it.wikipedia.org/wiki/Limite_notevole)

- $\lim_{N \rightarrow \infty} 1 - \frac{1}{N} = 1$

Solo il 37% degli slot svolge un lavoro utile. Pertanto, la velocità di trasmissione effettiva del canale non è  $R$  bps, ma solo 0.37  $R$  bps! Un'analisi simile mostra anche che il 37% degli slot va a vuoto e il 26% degli slot subisce collisioni.

# ALOHA puro

- "unslotted" ALOHA: più semplice, nessuna sincronizzazione
  - appena arriva un nuovo frame: lo trasmette immediatamente e integralmente
  - se la trasmissione va in collisione (lo rileva per esempio a causa dell'assenza di ACK): ritrasmette il frame immediatamente (dopo averne completato la trasmissione) con probabilità  $p$ , altrimenti attende il tempo di trasmissione di un frame e ripete il processo di attesa casuale, finché non ha successo
- la probabilità di collisione aumenta in assenza di sincronizzazione:
  - il frame inviato a  $t_0$  collide con altri frame inviati in  $[t_0-1, t_0+1]$



- Efficienza massima del protocollo ALOHA puro: 18% !

# Accesso multiplo con rilevamento della portante (carrier sense multiple access, CSMA)

**CSMA:** ascolta prima di trasmettere:

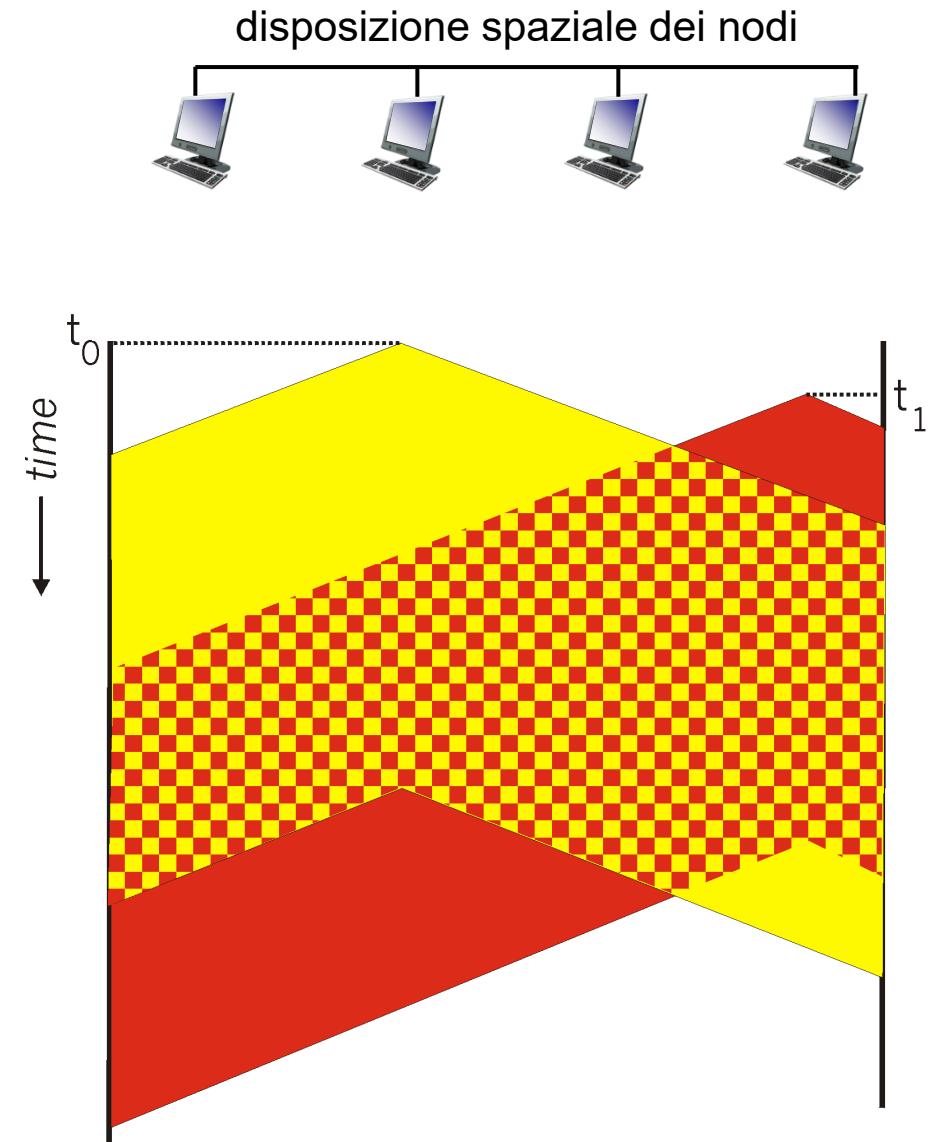
- se percepisce il canale inattivo: trasmette l'intero frame
- se percepisce il canale occupato: differisce la trasmissione
- analogia umana: non interrompere gli altri!

**CSMA/CD:** CSMA con *rilevamento della collisione (collision detection)*

- collisioni rilevate in breve tempo
- le trasmissioni in collisione vengono interrotte, riducendo gli sprechi di canale
- rilevamento delle collisioni facile con il cavo, difficile con il wireless
- analogia umana: se qualcun altro comincia a parlare insieme a voi, smettere di parlare

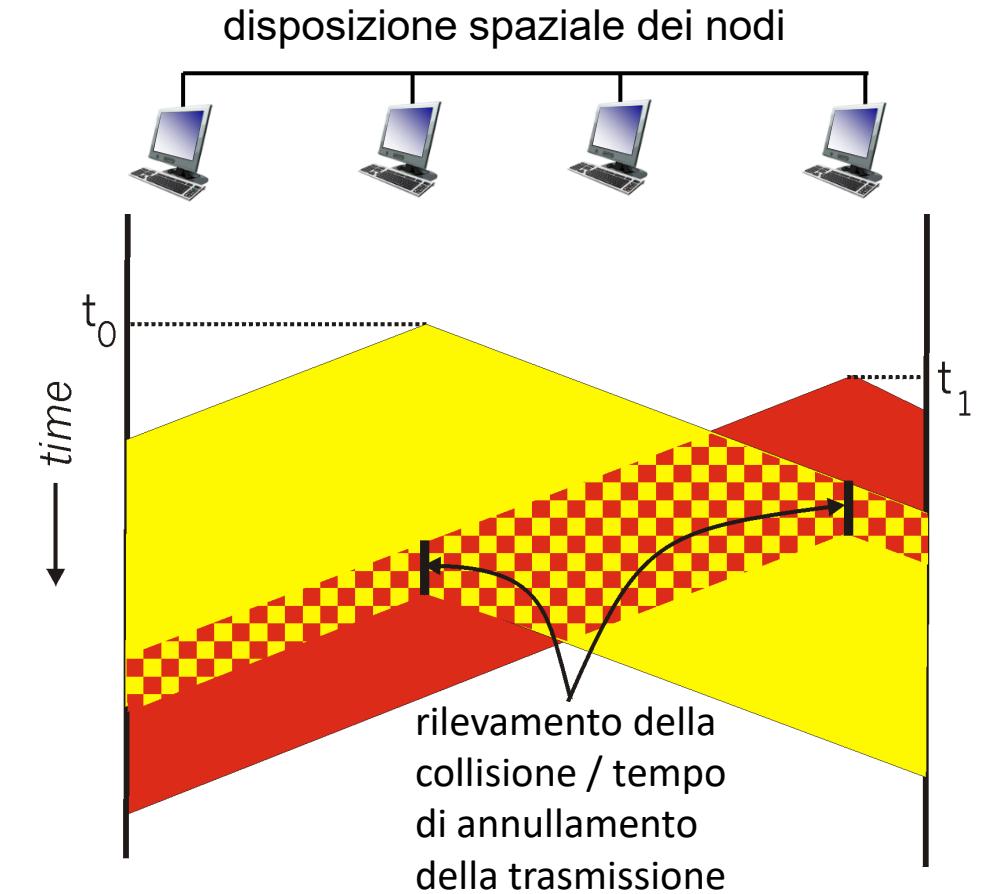
# CSMA: collisioni

- le collisioni possono *ancora* verificarsi con il rilevamento della portante:
  - il **ritardo di propagazione** significa che due nodi potrebbero non sentire la trasmissione appena avviata dell'altro
- **collisione:** spreco dell'intero tempo di trasmissione dei pacchetti
  - la distanza e il ritardo di propagazione giocano un ruolo importante nel determinare la probabilità di collisione



# CSMA/CD:

- CSMA/CD riduce la quantità di tempo sprecato nelle collisioni
  - trasmissione interrotta su rilevamento di collisione

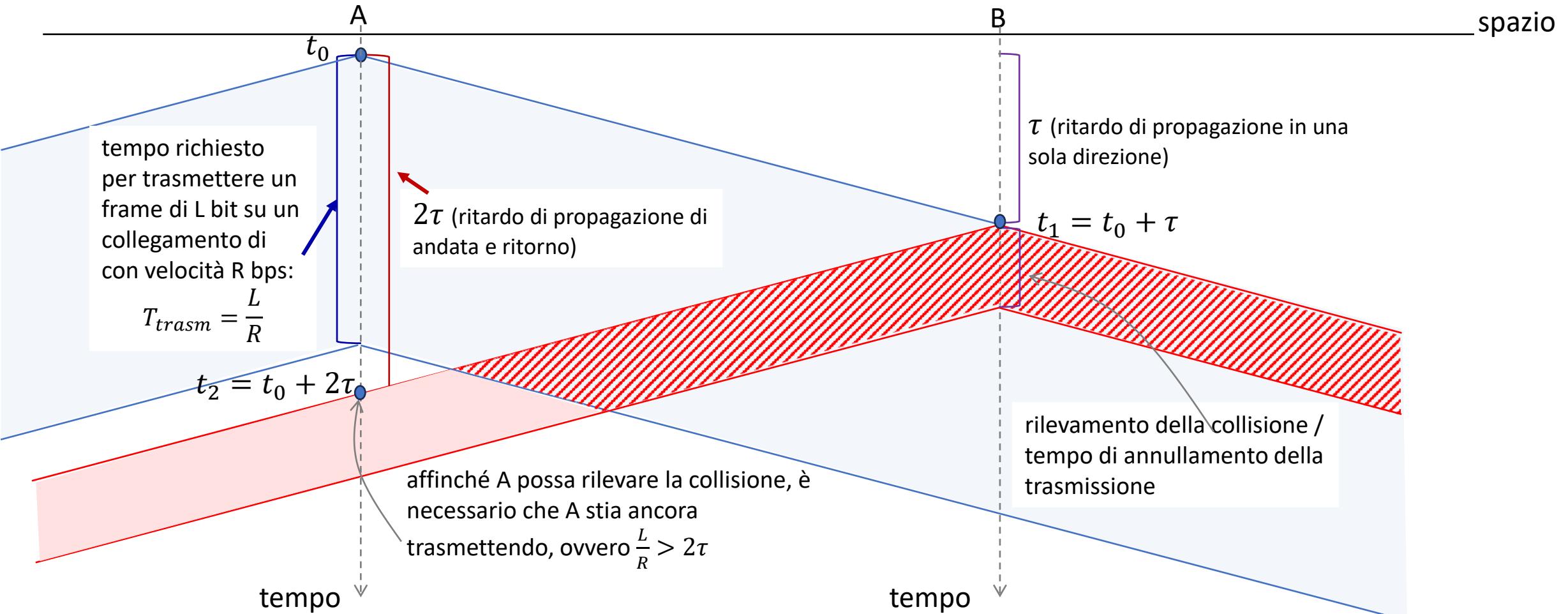


# Algoritmo CSMA/CD di Ethernet

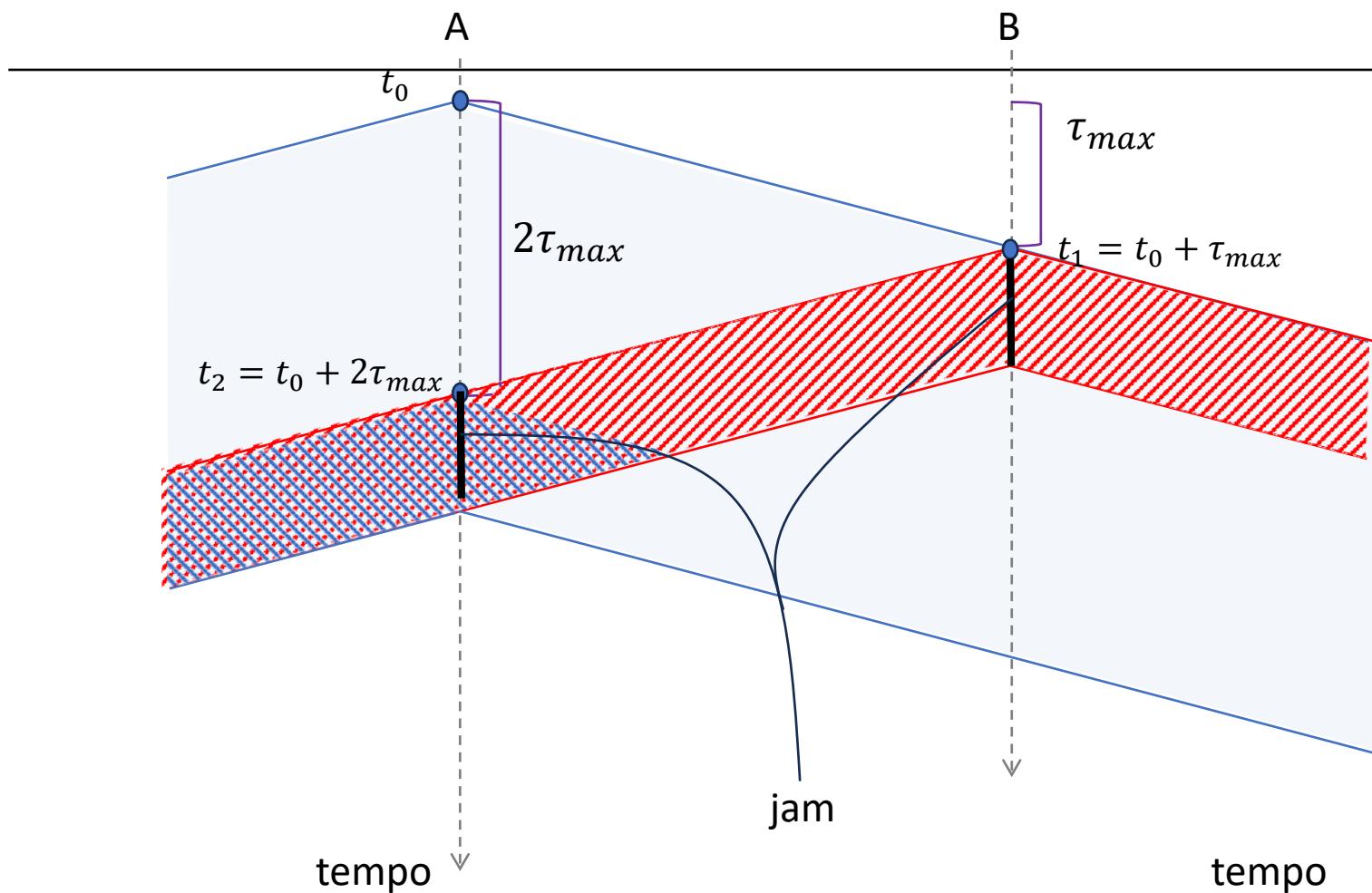
1. Ethernet riceve un datagramma dal livello di rete, crea un frame
2. se Ethernet ascolta il canale:
  - inutilizzato: avvia la trasmissione del frame.
  - occupato: aspetta finché il canale è libero, poi trasmette
3. se l'intero frame viene trasmesso senza collisioni - fatto!
4. se durante l'invio viene rilevata un'altra trasmissione: interrompere, inviare il segnale di disturbo (*jam*)
5. dopo aver interrotto, entra nella *binary exponential backoff*:
  - dopo la  $m$ -esima collisione, scegli  $K$  casualmente tra  $\{0, 1, 2, \dots, 2^m-1\}$ . Ethernet aspetta il tempo di trasmissione di  $K \cdot 512$  bit, ritorna allo Step 2
  - più collisioni: maggiore intervallo di backoff (ma  $m$  viene limitato a 10)

# CSMA/CD: Vincoli (1)

**B** inizia a trasmettere un attimo prima che possa rilevare il segnale di **A**. Rilevata la collisione (all'istante  $t_1 = t_0 + \tau$ ), **B** interrompe la trasmissione del frame e invia il disturbo (*jam*). Il segnale di **B** può essere rilevato da **A** all'istante  $t_2 = t_0 + 2\tau$ . Se A avesse già finito la trasmissione del suo frame, non rileverebbe la collisione.



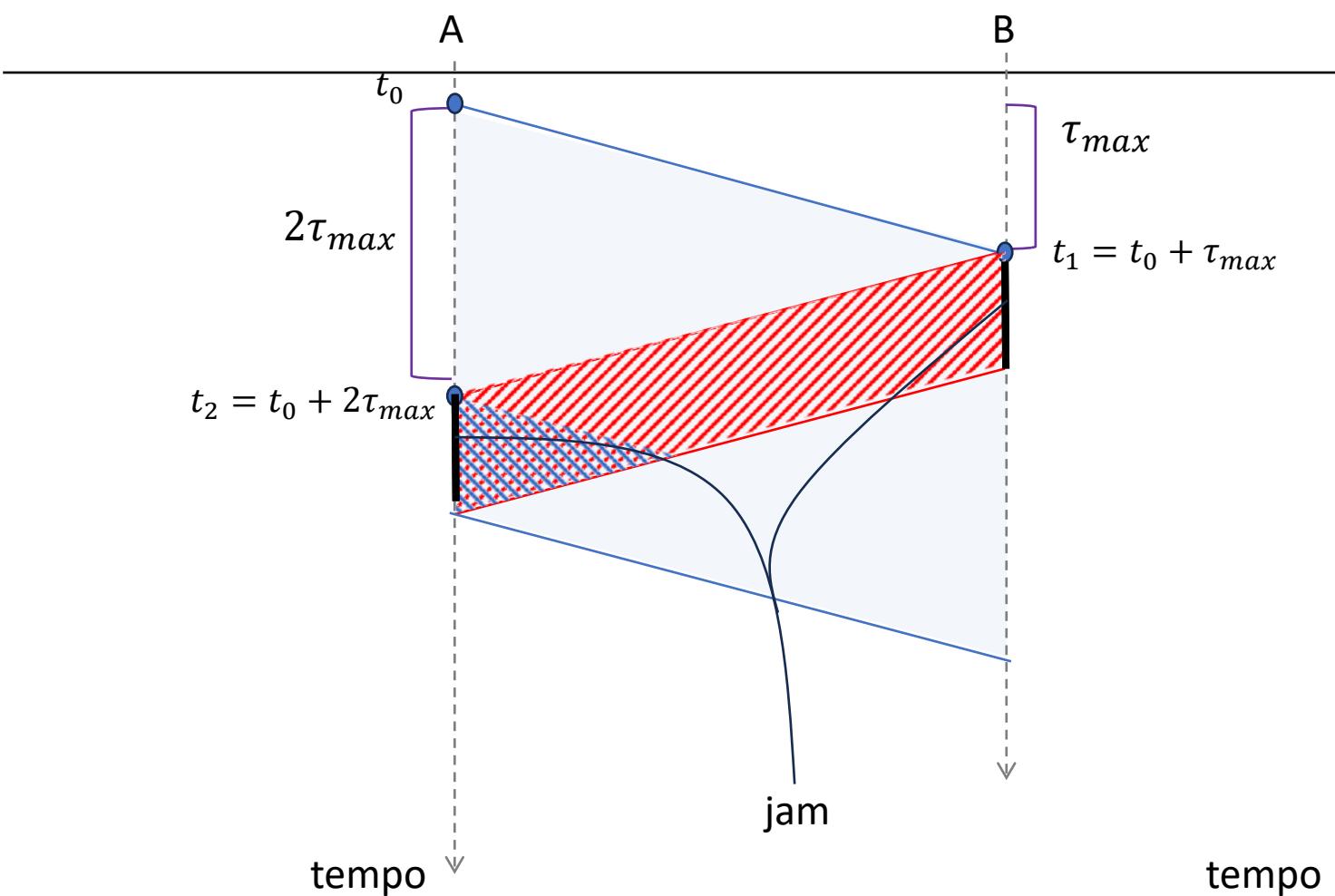
# CSMA/CD: Vincoli (2)



Viene stabilito un ritardo massimo  $\tau_{max}$ , che include il ritardo di propagazione attraverso il mezzo così come il contributo di altri componenti e processi, più un margine di sicurezza.

$\tau_{max}$  è spesso espresso in bit e va interpretato come il tempo impiegato per trasmettere quel numero di bit: ne consegue che maggiore è la velocità di trasmissione, minore è il ritardo ammesso in secondi. Ne deriva, quindi, che anche il limite di distanza fisica (il diametro fisico del dominio di collisione) diminuisce al crescere della velocità di trasmissione.

# CSMA/CD: Vincoli (3)



Supponiamo che **B** sia proprio alla massima distanza da **A**. Affinché CSMA/CD funzioni, cioè che **A** possa rilevare sempre la collisione, dobbiamo garantire che **A** stia ancora trasmettendo quando arriva il segnale da **B**. A tale scopo si richiede che un frame abbia una lunghezza minima tale per cui la sua trasmissione richieda almeno uno *slot time*, così definito:

$$\text{slot time} = 2\tau_{max} + \text{massima\_durata\_jam}$$

(uguale a 512 bit in Ethernet a 10 e 100 Mbps, cioè 64 byte)

Si noti che nel caso peggiore **B** inizia a trasmettere un *epsilon* prima di poter rilevare la trasmissione di **A**: quest'ultimo rileverà la collisione entro lo *slot time*. Trascorso uno slot time, **A** è sicuro che non rileverà più alcuna collisione, perché ciò richiederebbe che **B** abbia iniziato a trasmettere in un istante  $t'_1 > t_0 + \tau_{max}$ , cosa impossibile perché a quel punto **B** avrebbe potuto rilevare la trasmissione di **A** (carrier sense).

Si dice che **A** ha acquisito il canale.

# Efficienza CSMA/CD

- $d_{prop}$  = massimo ritardo di propagazione tra due schede di rete
- $d_{trasm}$  = tempo necessario per trasmettere un frame di dimensione massima

$$\text{efficienza} = \frac{1}{1 + 5d_{prop}/d_{trasm}}$$

- l'efficienza tende a 1
  - se  $t_{prop}$  tende a 0 (perché la trasmissione viene interrotta subito in presenza di collisioni, evitando sprechi)
  - se  $t_{trasm}$  tende a infinito (perché un frame, appropriatosi del canale, lo impegnava a lungo)
- prestazioni migliori di ALOHA: e semplice, economico, decentralizzato!

# Protocolli a rotazione

## protocolli a suddivisione del canale:

- condividere il canale in modo *efficiente* ed *equo* con un carico elevato
- inefficiente a basso carico: ritardo nell'accesso al canale (nel TDMA),  $1/N$  della larghezza di banda allocata anche se solo 1 nodo attivo!

## protocolli ad accesso casuale

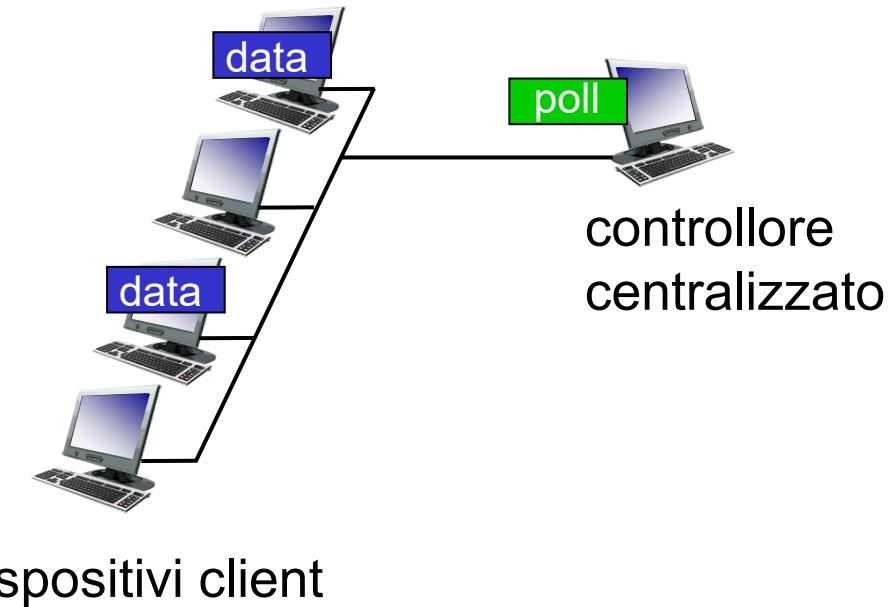
- efficiente a basso carico: un singolo nodo può usare il canale completamente
- alto carico: *overhead* di collisione

## protocolli a rotazione

- cercate il meglio dei due mondi!

# Protocolli a rotazione polling:

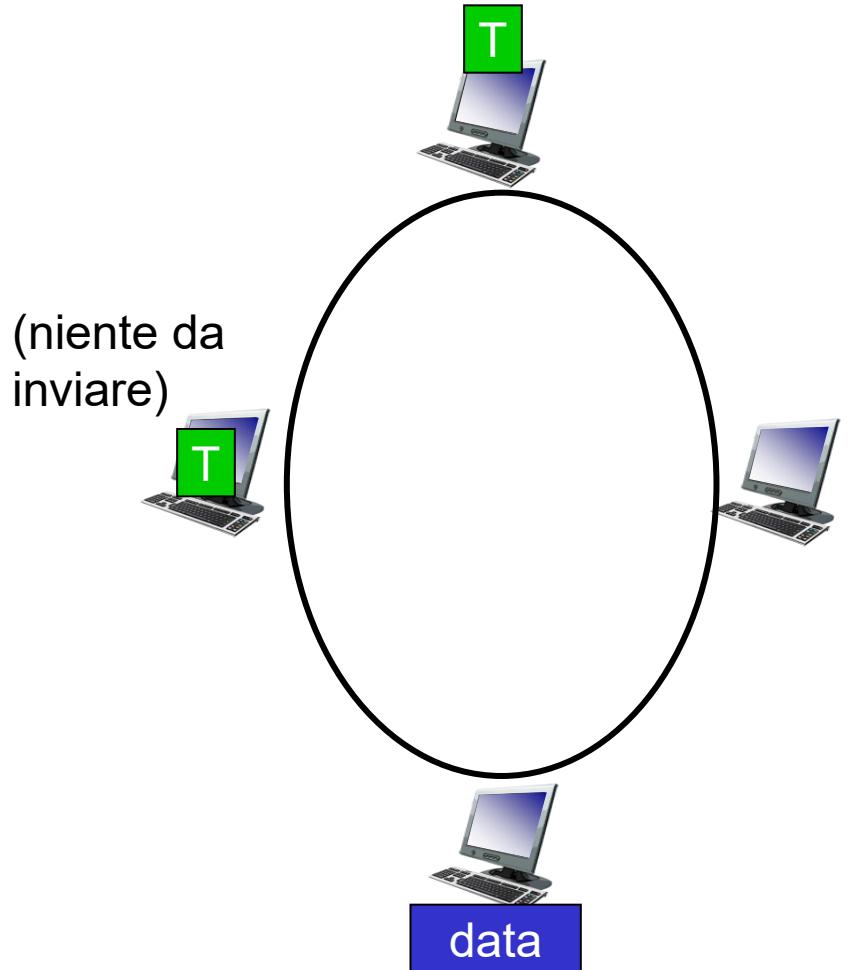
- il controllore centralizzato “invita” gli altri nodi a trasmettere a loro volta (fino a un massimo di frame per turno)
  - il controllore determina che il client ha finito osservando la mancanza di segnale
  - elimina collisioni e slot inutilizzati
- problemi:
  - **overhead del polling**
  - **ritardo di polling**: il tempo impiegato per notificare a un nodo il permesso di trasmettere -> anche in presenza di un solo nodo attivo, il controllore deve contattare periodicamente tutti gli altri nodi, determinando un throughput effettivo minore di R
  - **singolo punto di rottura** (master)
  - Il Bluetooth usa il polling



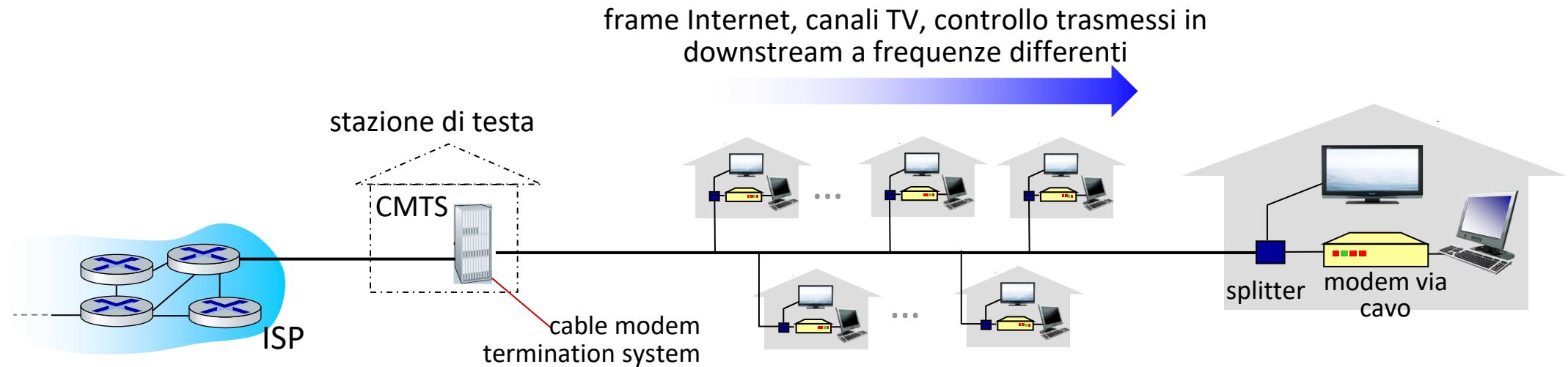
# Protocolli a rotazione

## token passing:

- Messaggio di controllo detto **token** (**gettone**) passato esplicitamente da un nodo al successivo, sequenzialmente
  - trasmette mentre possiede il token (entro un massimo accordato)
- problemi:
  - overhead associato al token
  - latenza
  - singolo punto di rottura (token)
  - Usato in: FDDI e token ring (IEEE 802.5)

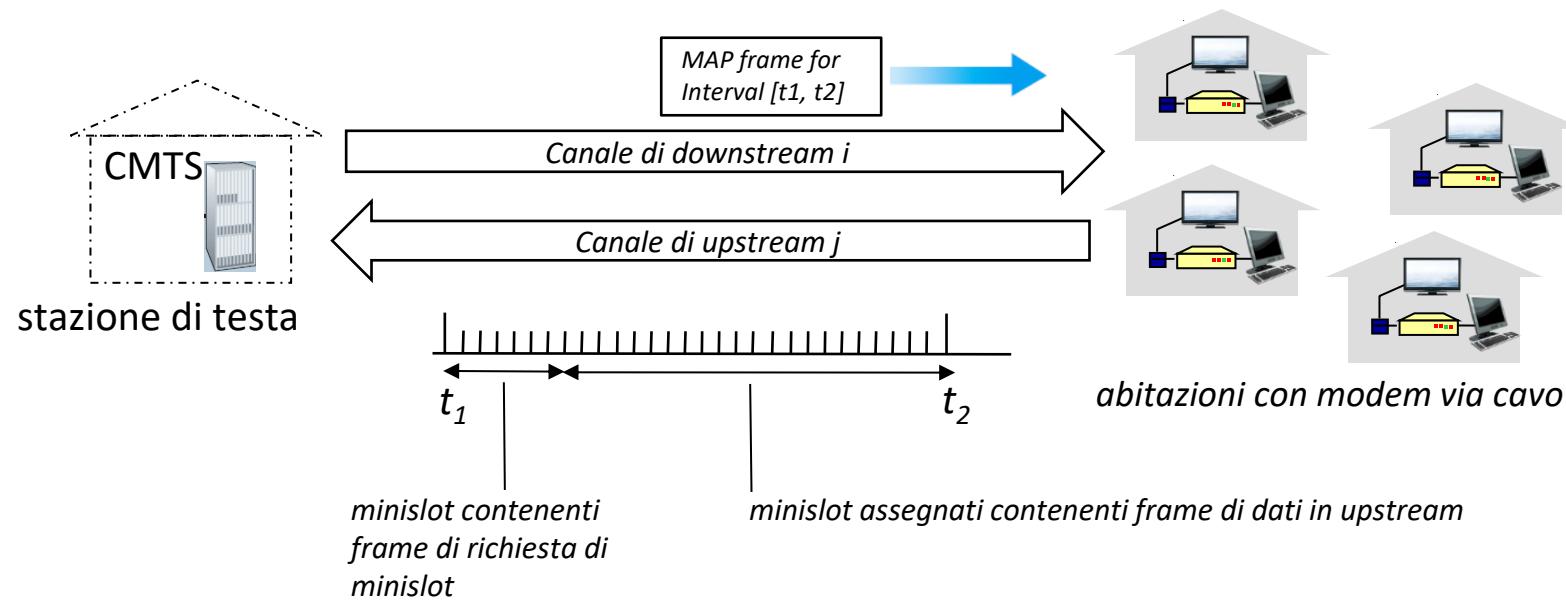


# Rete di accesso via cavo: FDM, TDM, allocazione centralizzata e accesso casuale!



- **molteplici** canali FDM downstream (broadcast): fino a 1.6 Gbps/canale
  - un solo CMTS trasmette nei canali -> nessun problema di accesso multiplo
- **molteplici** canali upstream (fino a 1 Gbps/canale)
  - **accesso multiplo**: tutti gli utenti si contendono (accesso casuale) determinati slot temporali del canale upstream; agli altri vengono assegnati TDM

# Rete di accesso via cavo;



**DOCSIS:** specifiche di interfaccia del servizio dati via cavo

- FDM su canali di frequenze upstream e downstream
- TDM upstream: alcuni slot assegnati, alcuni sono contesi
  - Frame MAP in downstream: assegna i minislot in upstream
  - Richieste di frame in upstream (e dati) trasmessi con accesso casuale (binary backoff) in slot selezionati

# Riassunto dei protocolli di accesso multiplo

- suddivisione del canale, per tempo, frequenza o codice
  - Time Division, Frequency Division
- accesso casuale (dinamico),
  - ALOHA, S-ALOHA, CSMA, CSMA/CD
  - rilevamento della portante: facile in alcune tecnologie (cablate), difficile in altre (wireless)
  - CSMA/CD usato in Ethernet
  - CSMA/CA usato in 802.11
- a rotazione
  - polling da un sito centrale, token passing
  - Bluetooth, FDDI, token ring

Università degli Studi di Roma "Tor Vergata"  
Laurea in Informatica

Sistemi Operativi e Reti  
(modulo Reti)  
a.a. 2024/2025

# Livello di collegamento (parte3)

dr. Manuel Fiorelli

[manuel.fiorelli@uniroma2.it](mailto:manuel.fiorelli@uniroma2.it)

<https://art.uniroma2.it/fiorelli>

Basate sulle slide del libro di testo:

[https://gaia.cs.umass.edu/kurose\\_ross/ppt.php](https://gaia.cs.umass.edu/kurose_ross/ppt.php)

Introduction: 1-59

# Livello di collegamento e LAN: tabella di marcia

- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- **LAN**
  - **indirizzamento, ARP**
  - Ethernet
  - switch
  - VLAN
- canali virtuali: MPLS
- Reti dei data center



- un giorno nella vita di una richiesta web

# LAN

## Local Area Network (LAN)

Copre un'area limitata come un'abitazione, una scuola, un ufficio o un edificio (o gruppo di edifici vicini).

Due tecnologie principali:

- **Ethernet** (questa tecnologia è usata anche in altri ambiti): IEEE 802.3 (nome del working group dell'IEEE e della famiglia di standard)
- **Wi-Fi**: IEEE 802.11

# Indirizzi MAC

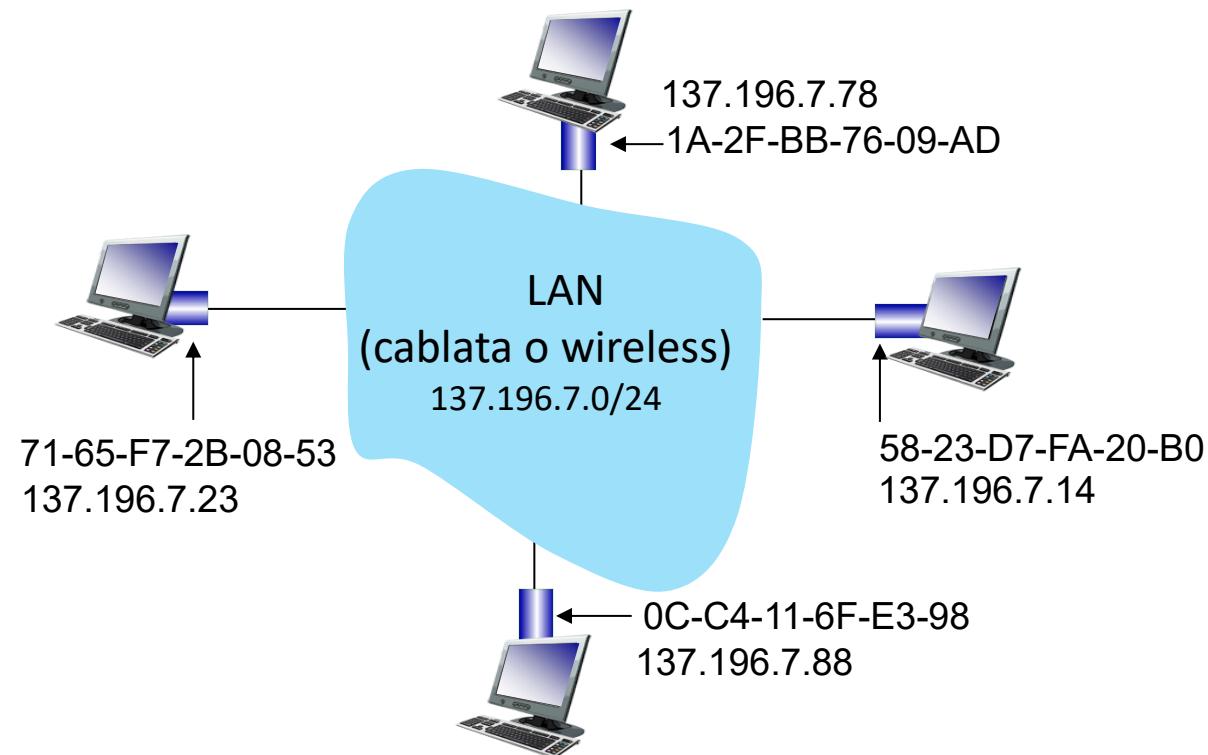
- indirizzi IP a 32 bit (128 bit in IPv6):
  - indirizzi *a livello di rete* per le interfacce
  - usati per l'inoltro a livello 3 (livello di rete)
  - es.: 128.119.40.136  
2001:**0**df0:**00**f2:**0000**:**0000**:**0000**:**0f10** → 2001:df0:f2::f10
- Indirizzi MAC (o LAN o fisici o Ethernet):
  - funzione: utilizzati “localmente” per portare i frame da un'interfaccia a un'altra interfaccia fisicamente connessa (stessa sottorete, nel senso dell'indirizzamento IP)
  - indirizzo MAC a 48 bit (per la maggior parte delle LAN) memorizzato nella ROM della NIC, a volte impostabile via software.
  - es.: 1A-2F-BB-76-09-AD

*notazione esadecimale (base 16)  
(ciascuna “cifra” rappresenta 4 bit)*

# Indirizzi MAC

ciascuna interfaccia in una LAN

- ha un indirizzo **MAC** univoco
- ha un indirizzo IP univoco (come abbiamo visto)

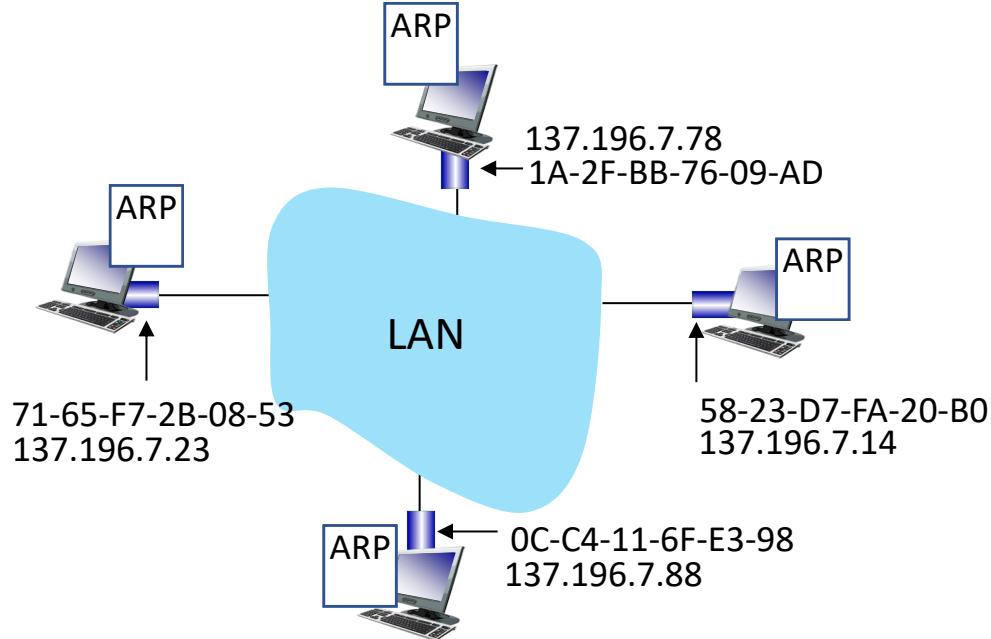


# Indirizzi MAC

- allocazione degli indirizzi MAC gestita dall'IEEE
- i produttori (di schede di rete) comprano porzioni dello spazio di indirizzi MAC (per assicurare l'unicità)
- analogia:
  - indirizzi MAC: come il codice fiscale
  - indirizzo IP: come l'indirizzo postale
- indirizzo MAC (piatto): portabilità
  - è possibile spostare un'interfaccia da una LAN a un'altra
  - indirizzo IP (gerarchico) *non* portabile: dipende dalla sottorete IP alla quale il nodo è connesso

# Protocollo per la risoluzione degli indirizzi (*address resolution protocol*, ARP)

**Domanda:** come determinare l'indirizzo MAC di un'interfaccia, conoscendo il suo indirizzo IP?



**Tabella ARP:** ogni nodo IP (host, router) sulla LAN ha una tabella (una per ciascuna interfaccia)

- corrispondenza tra indirizzi IP e MAC per alcuni nodi sulla LAN:  
<indirizzo IP; indirizzo MAC address; TTL>
- TTL (Time To Live): tempo dopo il quale la mappatura degli indirizzi sarà dimenticata (in genere 20 min da quando la voce è stata inserita nella tabella)

# Protocollo ARP in azione

esempio: A vuole inviare un datagramma a B

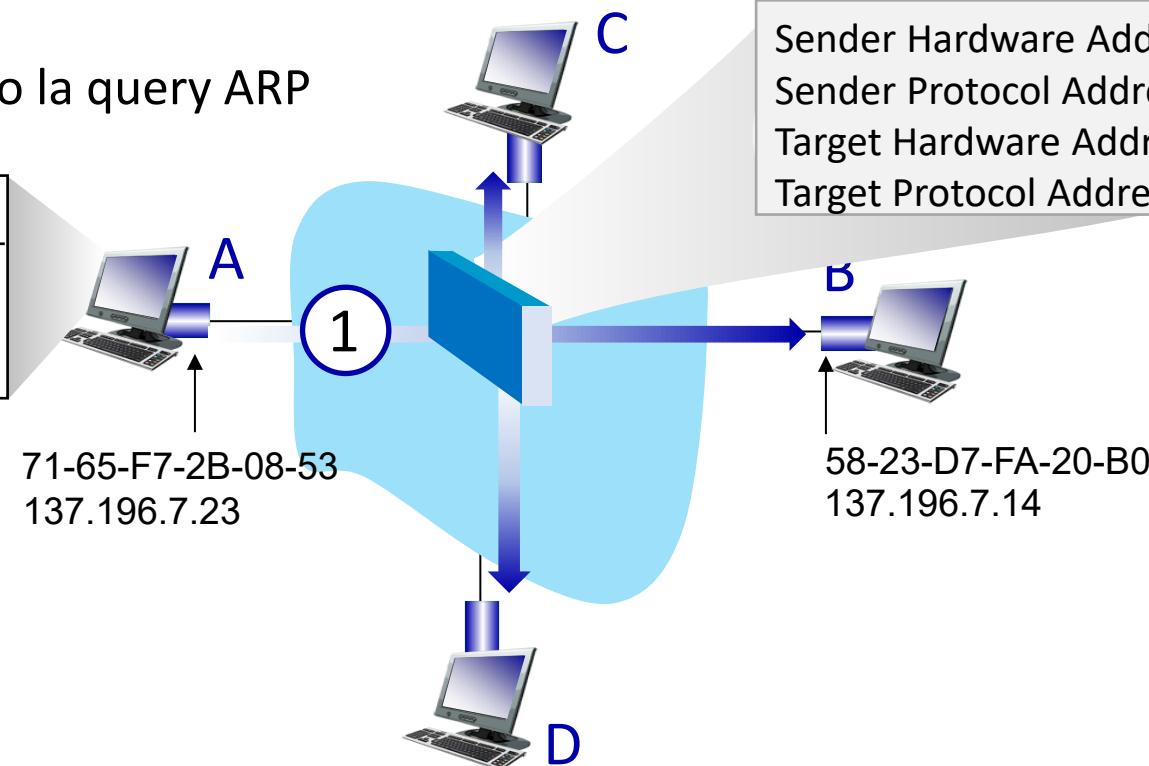
- l'indirizzo MAC di B non è nella tabella ARP di A, pertanto A usa ARP per trovare l'indirizzo MAC di B

A invia in broadcast una richiesta ARP,

- 1) contenente l'indirizzo IP di B
- indirizzo MAC di destinazione = FF-FF-FF-FF-FF-FF
  - tutti i nodi sulla LAN ricevono la query ARP

Tabella ARP in A

IP addr	MAC addr	TTL



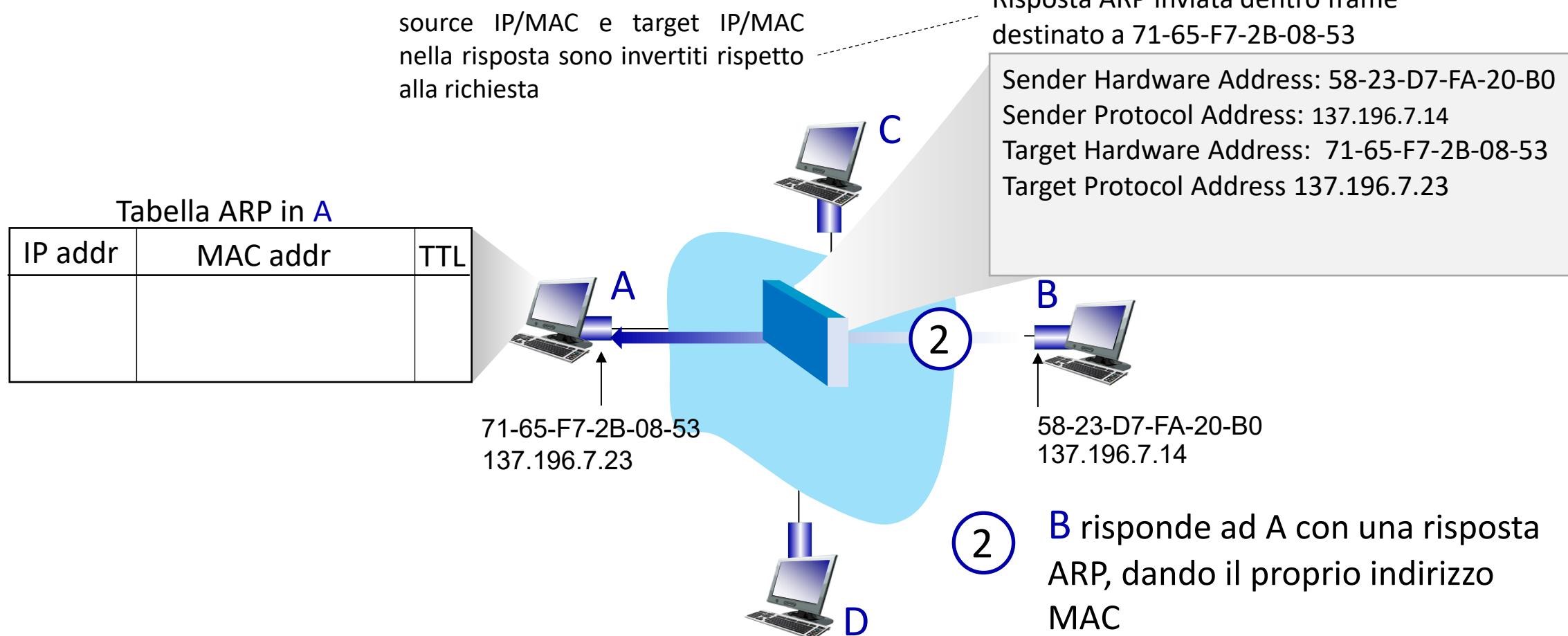
Richiesta ARP inviata dentro frame destinato a FF-FF-FF-FF-FF-FF

Sender Hardware Address: 71-65-F7-2B-08-53  
Sender Protocol Address 137.196.7.23  
Target Hardware Address: 00-00-00-00-00-00  
Target Protocol Address: 137.196.7.14

# Protocollo ARP in azione

esempio: A vuole inviare un datagramma a B

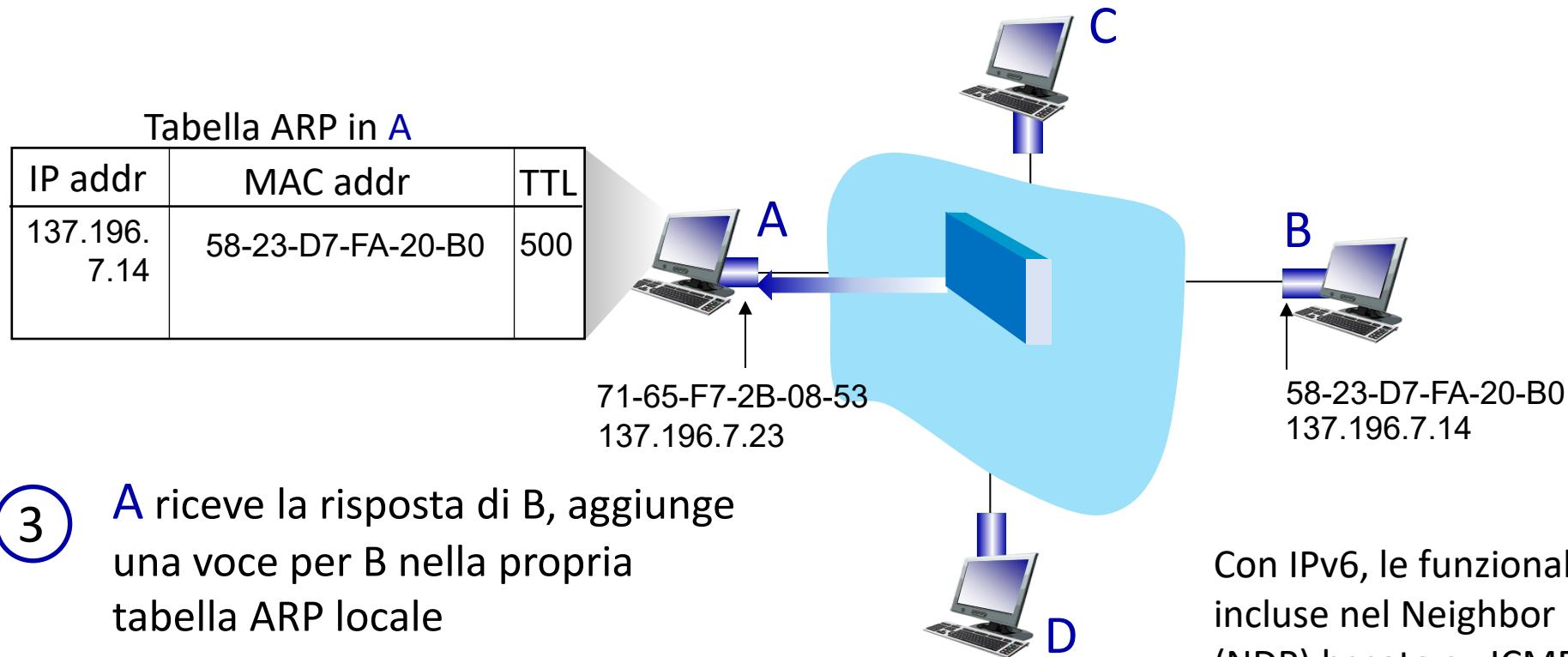
- L'indirizzo MAC di B non è nella tabella ARP di A, pertanto A usa ARP per trovare l'indirizzo MAC di B



# Protocollo ARP in azione

esempio: A vuole inviare un datagramma a B

- L'indirizzo MAC di B non è nella tabella ARP di A, pertanto A usa ARP per trovare l'indirizzo MAC di B



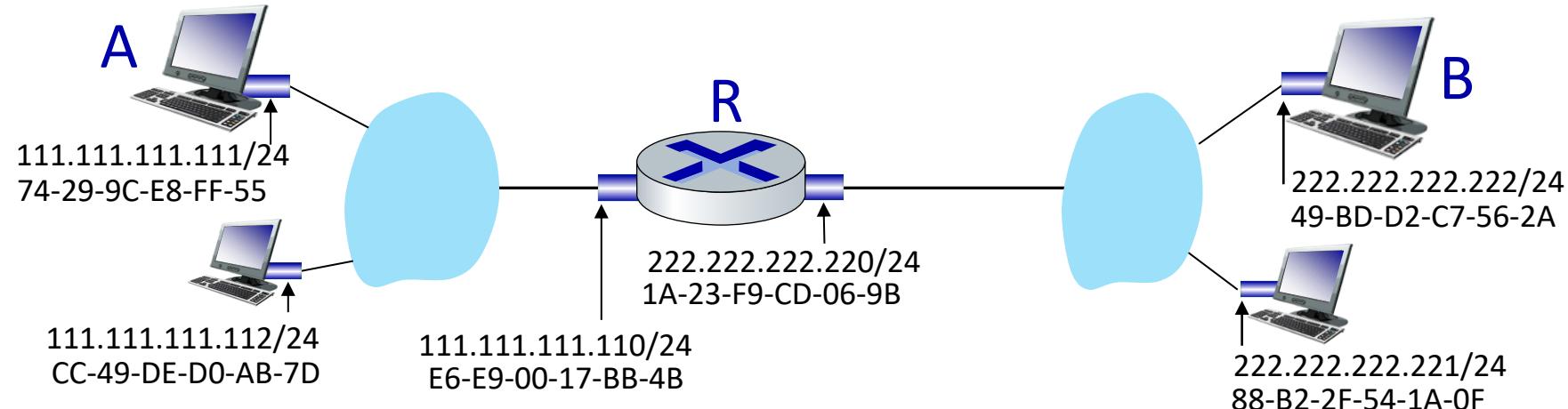
# ARP Spoofing o ARP Poisoning

- Un attaccante invia in una LAN risposte ARP contraffatte, inducendo l'associazione di un indirizzo IP a un certo indirizzo MAC
- Il protocollo ARP è senza stato e un nodo (host o router) aggiorna la propria ARP appena viene ricevuta una risposta ARP (a prescindere che questa faccia seguito a una effettiva richiesta)
- Alcuni "usi":
  - *denial-of-service* (DoS): associare diversi indirizzi IP allo stesso indirizzo MAC per sovraccaricarlo di traffico
  - *man-in-the-middle* (MITM): l'attaccante associa il proprio indirizzo MAC all'indirizzo IP di un altro nodo, in modo da intercettare (e magari modificare) il traffico destinato a quest'ultimo, per poi re-inoltrarglielo

# Come inviare un datagramma a un nodo esterno alla sottorete

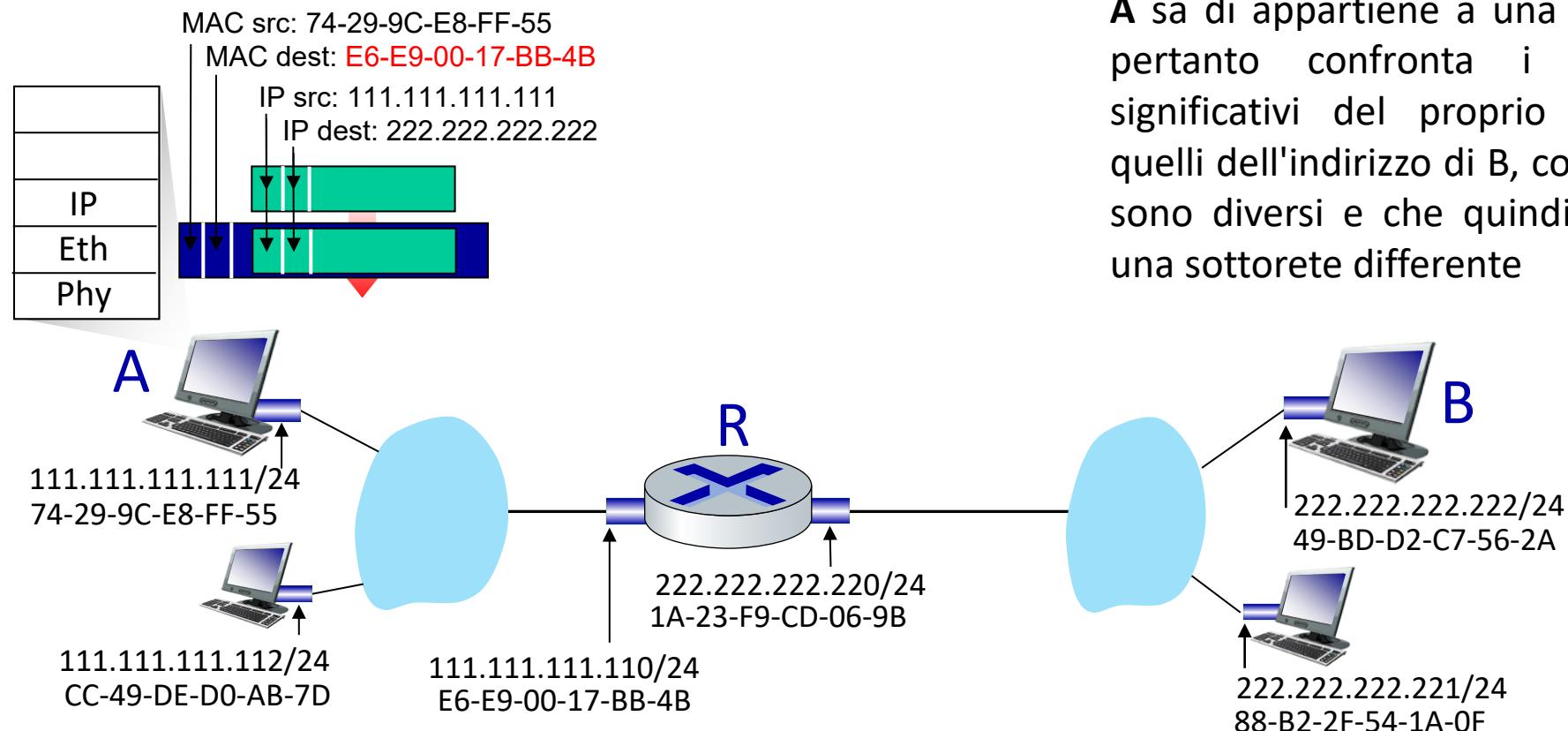
scenario dettagliato: **invio di un datagramma da A a B passando per R**

- attenzione sugli indirizzi – a livello IP (datagramma) e MAC (frame)
- assunzioni:
  - A conosce l'indirizzo IP di B
  - A conosce l'indirizzo IP dell'interfaccia di R nella propria sottorete (**come?** DHCP)
  - A conosce l'indirizzo MAC dell'interfaccia di R nella propria sottorete (**come?** ARP)



# Come inviare un datagramma a un nodo esterno alla sottorete

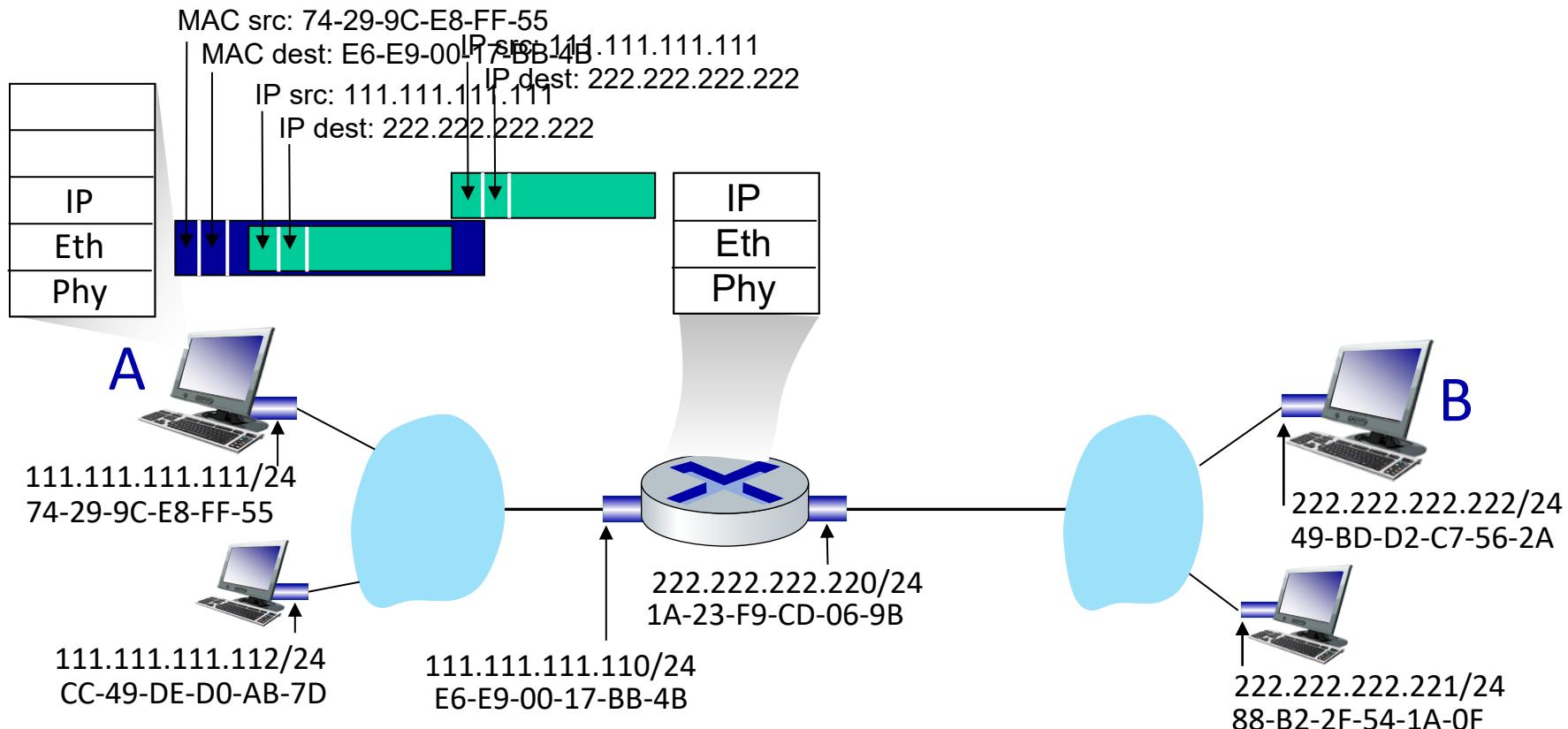
- A crea un datagramma IP con sorgente A e destinazione B
- A crea un frame a livello di collegamento contenente il datagramma IP da A a B
  - la destinazione del frame è l'indirizzo MAC di R



A sa di appartenere a una sottorete /24, pertanto confronta i 24 bit più significativi del proprio indirizzo con quelli dell'indirizzo di B, constatando che sono diversi e che quindi B si trova in una sottorete differente

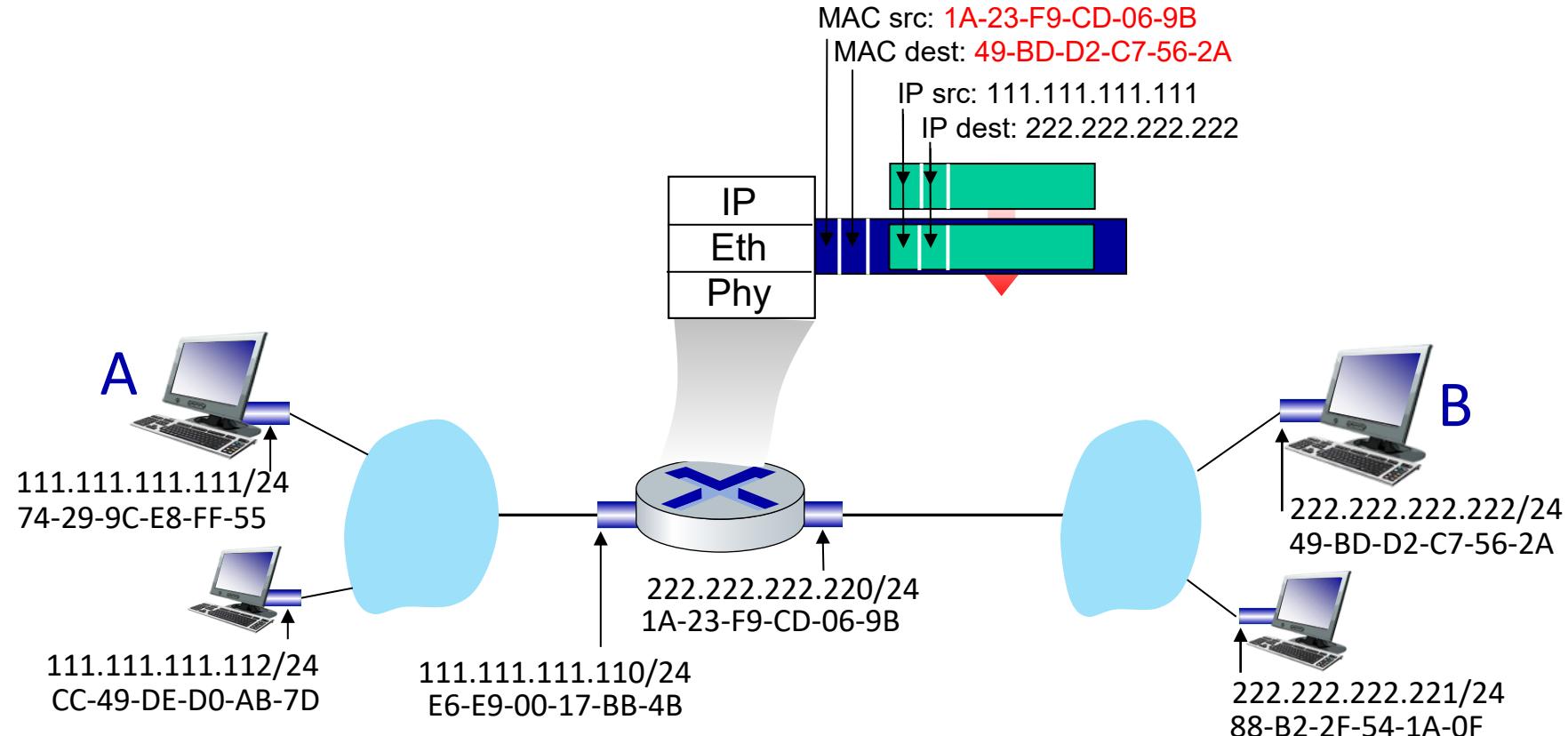
# Come inviare un datagramma a un nodo esterno alla sottorete

- frame inviato da A a R
- frame ricevuto da R, datagramma, passato in alto a IP



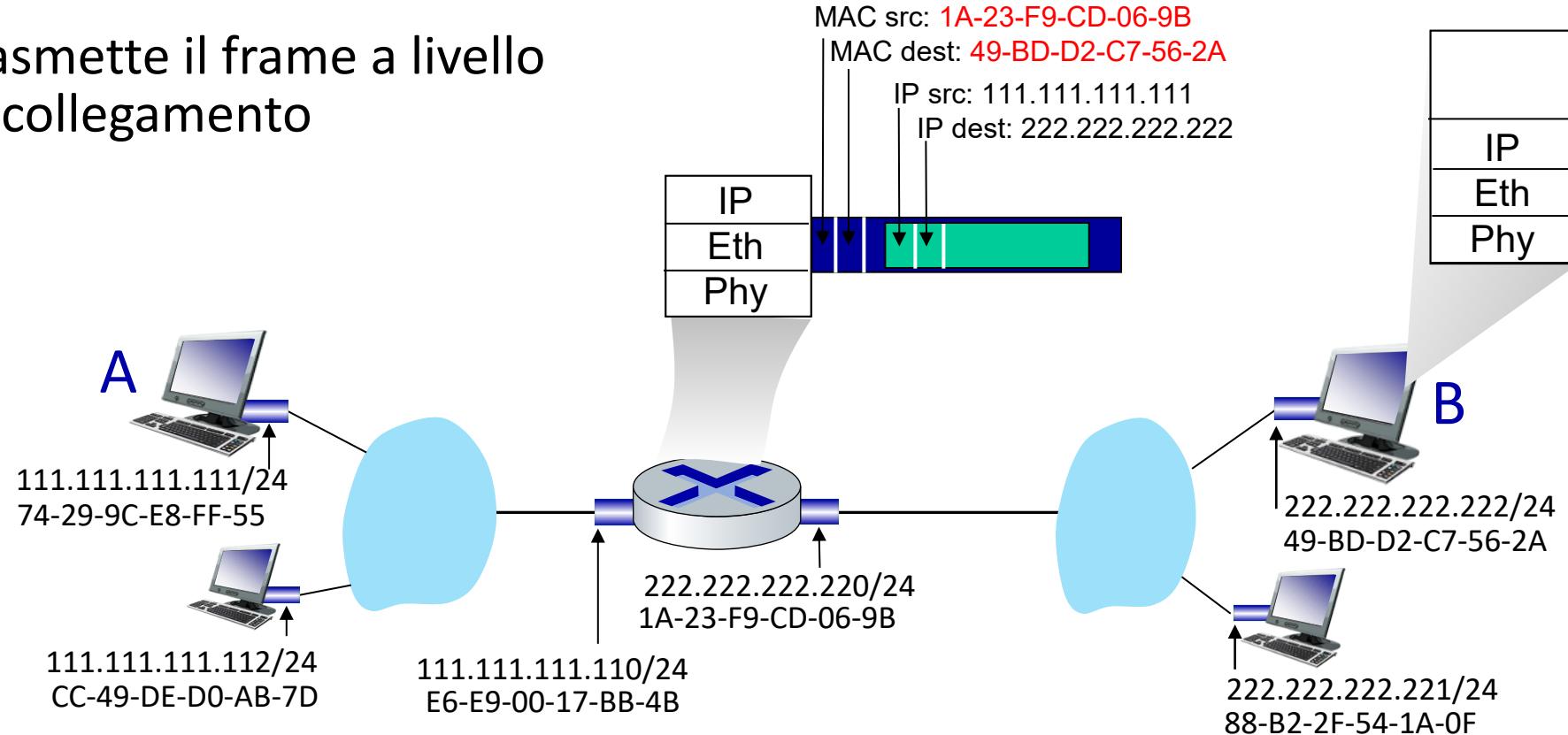
# Come inviare un datagramma a un nodo esterno alla sottorete

- R determina l'interfaccia di uscita, passa il datagramma con sorgente IP A e destinazione IP B al livello di collegamento
- R crea il frame a livello di collegamento contenente il datagramma IP da A a B. Indirizzo di destinazione del frame: indirizzo MAC di B



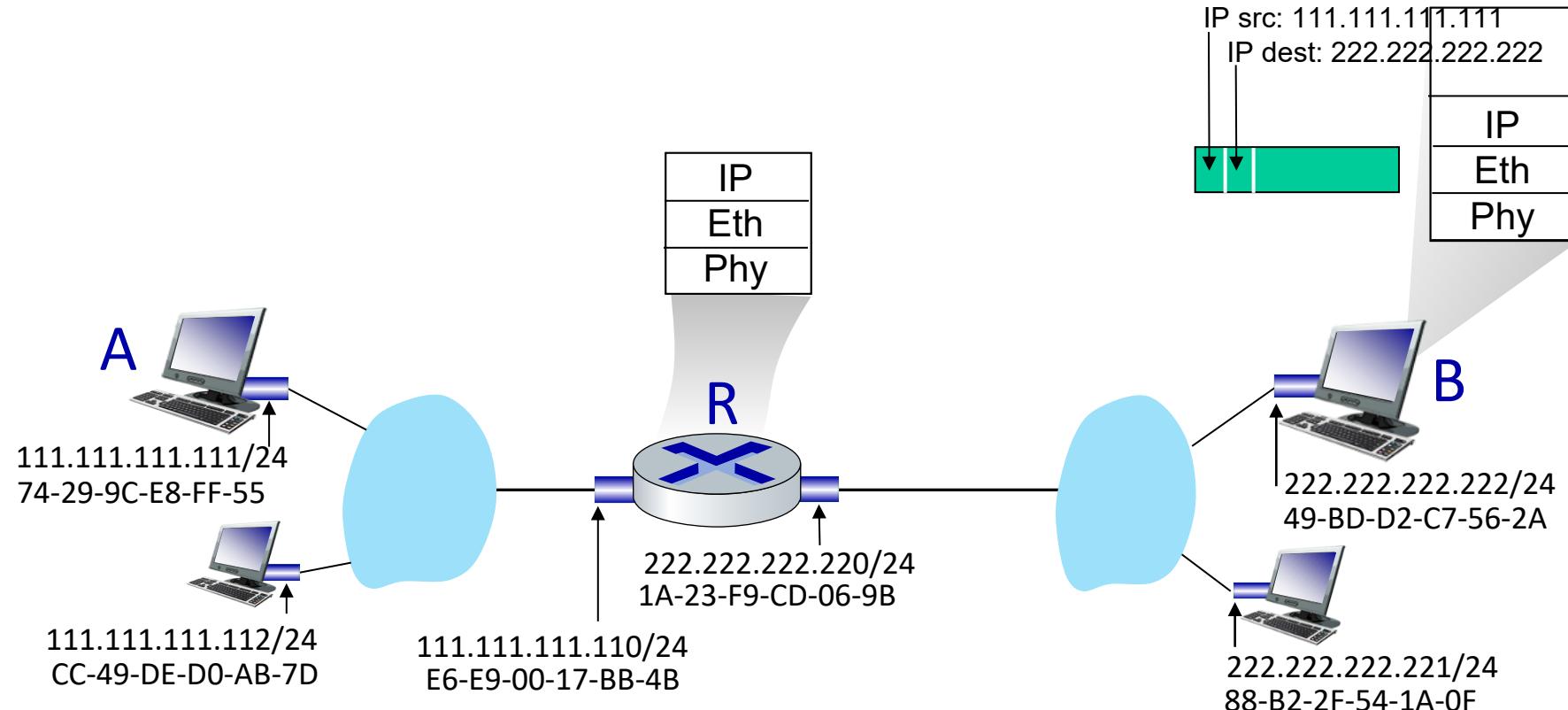
# Come inviare un datagramma a un nodo esterno alla sottorete

- R determina l'interfaccia di uscita, passa il datagramma con sorgente IP A e destinazione IP B al livello di collegamento
- R crea il frame a livello di collegamento contenente il datagramma IP da A a B. Indirizzo di destinazione del frame: indirizzo MAC di B
- trasmette il frame a livello di collegamento



# Come inviare un datagramma a un nodo esterno alla sottorete

- B riceve il frame, il datagramma IP destinato a sé
- B passa il datagramma in alto nella pila protocollare a IP



# Livello di collegamento e LAN: tabella di marcia

- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- **LAN**
  - indirizzamento, ARP
  - **Ethernet**
  - switch
  - VLAN
- canali virtuali: MPLS
- Reti dei data center



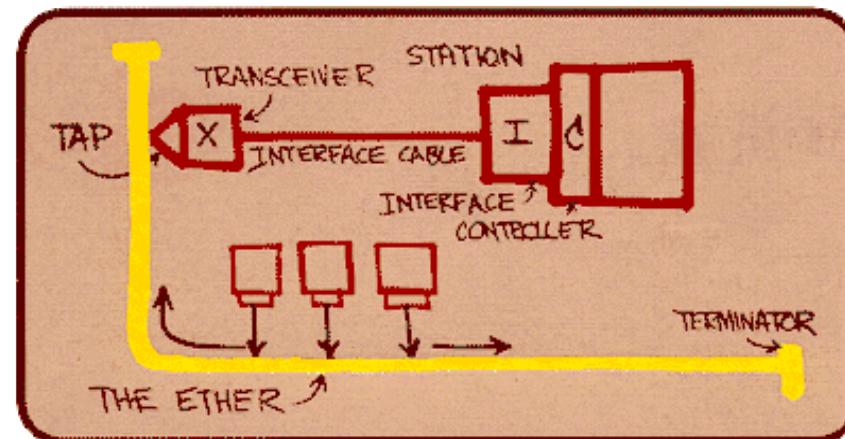
- un giorno nella vita di una richiesta web

# Ethernet

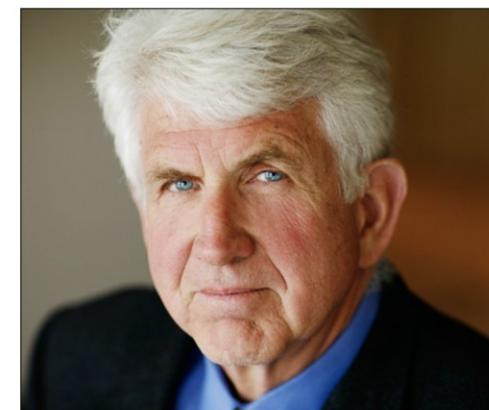
Tecnologia “dominante” per le LAN cablate:

- prima tecnologia LAN ampiamente utilizzata
- semplice, economica
- ha tenuto il passo sulla velocità: 10 Mbps – 400 Gbps
- singolo chip, più velocità (es., Broadcom BCM5761)

*Schizzo Ethernet di Metcalfe*



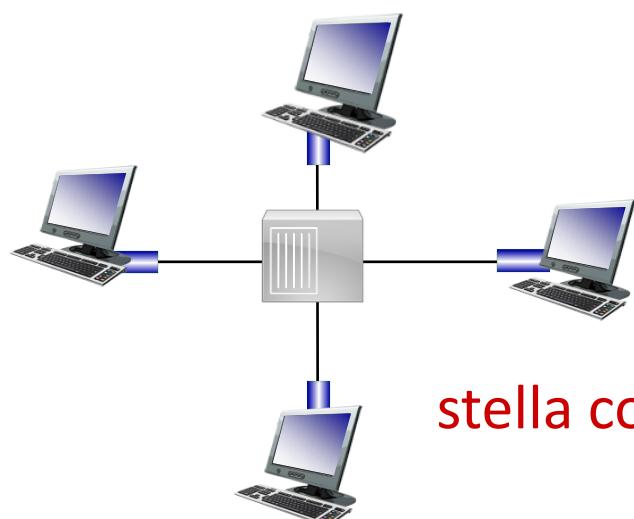
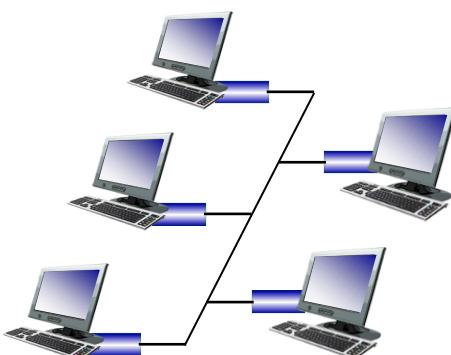
Bob Metcalfe: co-inventore di Ethernet,  
destinatario del Premio ACM Turing 2022



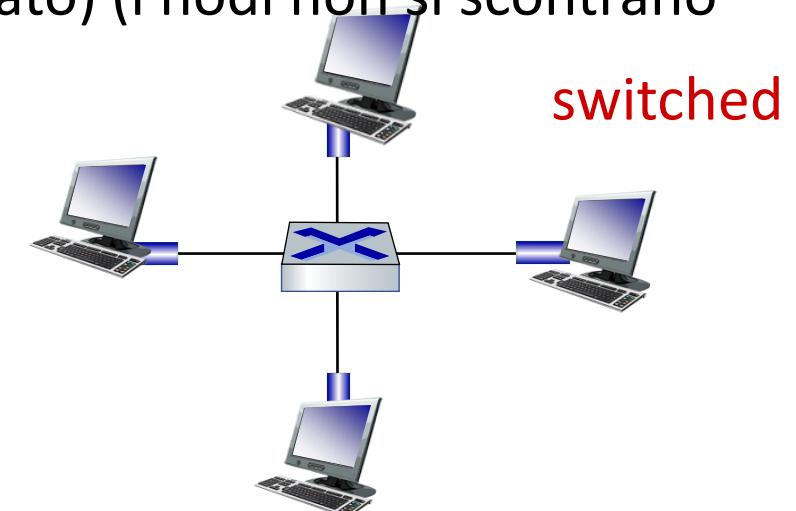
# Ethernet: topologia fisica

- **bus:** popolare fino alla metà degli anni '90
  - tutti i nodi sono nello stesso dominio di collisione (possono collidere tra loro)
- **topologia a stella con hub:** popolare fino agli anni 2000
  - i nodi sono interconnessi da un hub (dispositivo a livello fisico che rigenera i segnali ricevuti su una interfaccia e li ritrasmette su tutte le altre interfacce), pertanto tutti i nodi sono nello stesso dominio di collisione
- **commutata (switched):** oggi prevalente
  - *switch* di livello 2 attivo al centro
  - ogni “spoke” esegue un protocollo Ethernet (separato) (i nodi non si scontrano tra loro)

**bus:** cavo coassiale



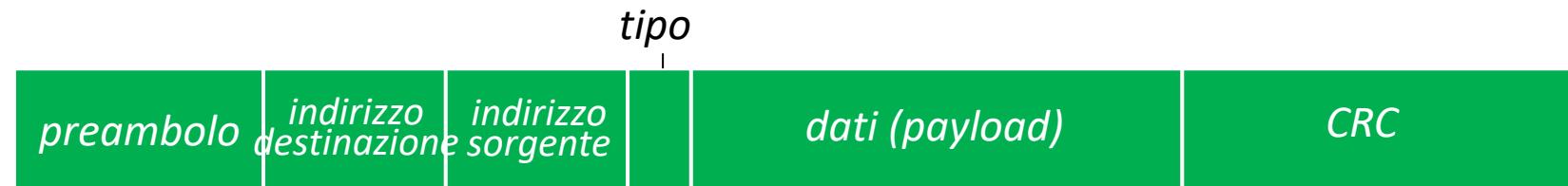
**stella con hub**



**switched**

# Struttura del frame Ethernet

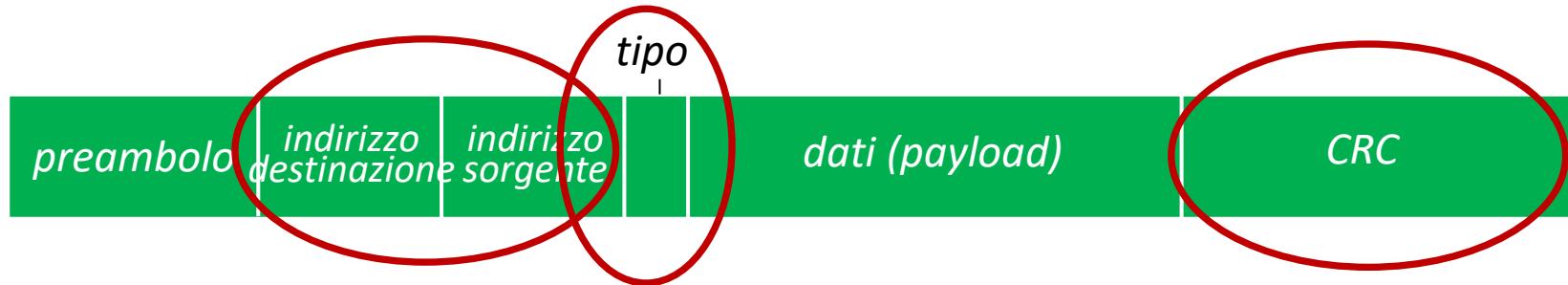
l'interfaccia trasmittente incapsula il datagramma IP (o altro pacchetto di protocolli di livello di rete) in **frame Ethernet**



## *preamble:*

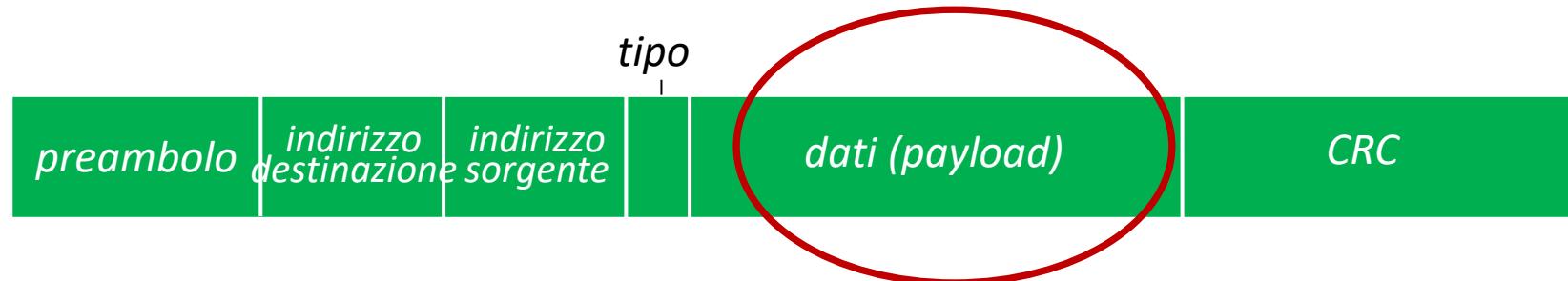
- usato per “risvegliare” le schede di rete dei riceventi e sincronizzare i loro clock con quello del trasmittente
- 7 byte di 10101010 seguiti da un byte di 10101011 questi due 1 consecutivi, che rompono il pattern di 1 e 0 alternati, informano il ricevente dell'inizio del frame vero e proprio

# Struttura del frame Ethernet (continuazione)



- **indirizzi**: indirizzi sorgente e destinazione a 6 byte
  - se l'adattatore riceve un frame con un indirizzo di destinazione corrispondente o con un indirizzo di broadcast (ad esempio, un pacchetto ARP), passa i dati nel frame al protocollo di livello di livello superiore
  - altrimenti, l'adattatore scarta il frame
- **tipo**: indica un protocollo di livello superiore (2 byte)
  - principalmente IP, ma sono possibili anche altri protocolli a livello di rete, ad es. Novell IPX, AppleTalk, ma anche ARP
  - utilizzato per demultiplexare sul ricevitore
- **CRC**: controllo di ridondanza ciclica presso il ricevitore (4 byte)
  - errore rilevato: il frame viene scartato

# Struttura del frame Ethernet (continuazione)



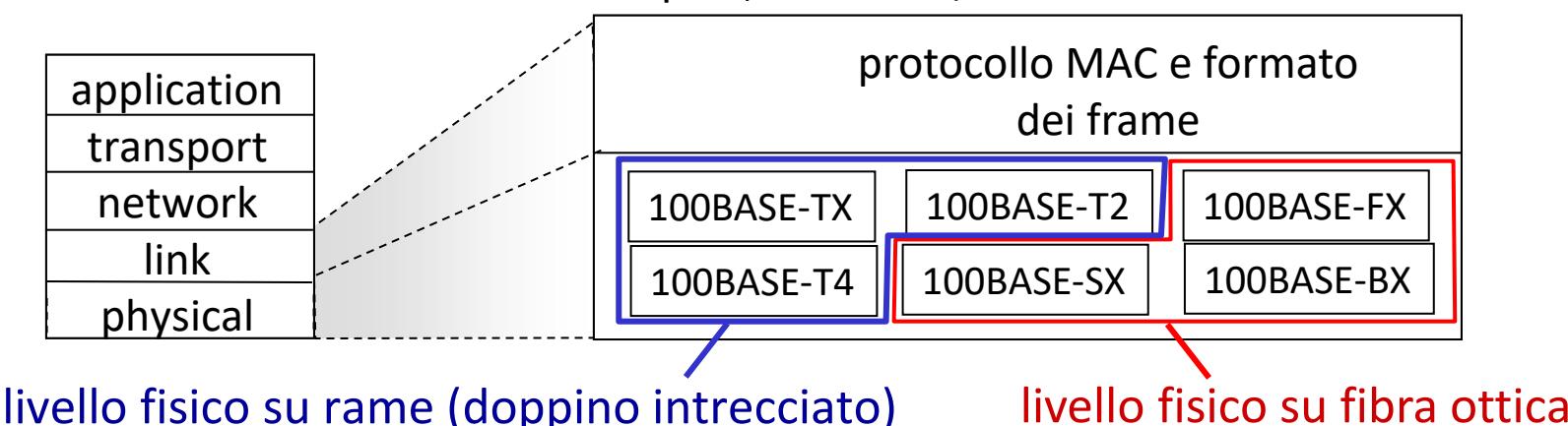
- **payload:** dati passati al protocollo di livello superiore
  - minimo 46 byte: se più piccolo deve essere aggiunto del padding; perciò è importante che il protocollo di livello superiore preveda un meccanismo per stabilire la reale dimensione dei dati
  - massimo 1500 byte (salvo estensioni) → MTU
- La **fine del frame** è determinata a livello fisico dall'assenza di transizioni sulla linea
- escludendo il preambolo, la **dimensione di un frame** è compresa tra 64 byte (512 bit) e 1518 byte → slot time 512 bit. In Gigabit Ethernet lo slot time è in realtà di 4096 bit (512 byte): frame più piccoli richiedono l'aggiunta di padding (fatto in maniera trasparente dall'hardware)

# Ethernet: non affidabile, senza connessione

- **senza connessione:** nessun handshake tra le NIC mittente e ricevente
- **non affidabile:** la NIC ricevente non invia ACK o NAK alla NIC mittente
  - i dati nei frame scartati vengono recuperati solo se il mittente iniziale utilizza un trasferimento dati affidabile di livello superiore (ad esempio, TCP), altrimenti i dati scartati vanno persi
- Protocollo MAC di Ethernet: "unslotted" **CSMA/CD con binary backoff**

# 802.3 Ethernet standard: livelli di collegamento e fisico

- *molti* standard Ethernet differenti
  - protocollo MAC e formato dei frame comuni
  - velocità differenti: 2 Mbps, ... 100 Mbps, 1Gbps, 10 Gbps, 40 Gbps, 100 Gbps
    - mezzi trasmissioni differenti: cavo coassiale, doppino, fibra
    - limiti di lunghezza:
      - segmento: 100 m su doppino intrecciato (Cat. 5) per Fast Ethernet (100 Mbps) e Gigabit Ethernet (1 Gbps); da qualche centinaio di metri fino a decine chilometri in base alla classe di fibra ottica
      - dominio di collisione: il ritardo round-trip deve essere inferiore allo slot time: occorre considerare il ritardo dovuto alla propagazione nel mezzo, così come quello introdotto da hub o repeater, dagli adattatore, etc.: ~2500 m 10BASE-T; ~ 205 m 100BASE-TX; Gigabit Ethernet solitamente usato in modalità full-duplex, no CSMA/CD



# System considerations for multisegment 100BASE-T networks

Table 29-3—Network component delays, Transmission System Model 2

Component	Round trip delay in bit times per meter	Maximum round trip delay in bit times
Two TX/FX DTEs		100
Two T4 DTEs		138
Two T2 DTEs		96
One T2 or T4 and one TX/FX DTE <sup>a</sup>		127
Cat 3 cabling segment	1.14	114 (100 m)
Cat 4 cabling segment	1.14	114 (100 m)
Cat 5 cabling segment	1.112	111.2 (100 m)
STP cabling segment	1.112	111.2 (100 m)
Fiber optic cabling segment	1.0	412 (412 m)
Class I repeater		140
Class II repeater with all ports TX/FX		92
Class II repeater with any port T4		67
Class II repeater with any port T2		90

2 cavi da 100 m, uno per ciascun host

margine di sicurezza

$$127 + 2 \cdot 111.2 + 140 + 4 = 493,4 < 512$$

Considero una rete Fast Ethernet con topologia a stella (con hub)

I ritardi sono espresso in *bit time*, ovvero il tempo impiegato per immettere un bit nel collegamento = 1/R dove R è la velocità di trasmissione.

Se R = 100 Mbps, 1 bit time =  $\frac{1}{100 \cdot 10^6} = 10^{-8} s$

<sup>a</sup> Worst-case values are used (TX/FX values for MAC transmit start and MDI input to collision detect; T4 value for MDI input to MDI output).

# System considerations for multisegment 100BASE-T networks

Che vuol dire che 100 m di cavo UTP di categoria 5 hanno un round trip delay di 111.2 bit time (rispetto a 100BASE-T)?

111.2 bit time corrispondono a  $111.2 \times 10^{-8}$ s. Parlando di round trip delay, si intende che in questo intervallo di tempo il segnale può attraversare il cavo avanti e indietro, percorrendo cioè 200 m.

Pertanto, la velocità di propagazione del cavo è  $v = \frac{200}{111.2 \times 10^{-8}} m/s = 1.799 \cdot 10^8 m/s$

Considerando la velocità della luce  $c = 299792458 m/s \approx 3 \cdot 10^8 m/s$   
 $\frac{v}{c} = 0,6$

Cioè la velocità di propagazione del segnale in questo mezzo è il 60% della velocità della luce.

# Livello di collegamento e LAN: tabella di marcia

- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- **LAN**
  - indirizzamento, ARP
  - Ethernet
  - **switch**
  - VLAN
- canali virtuali: MPLS
- Reti dei data center



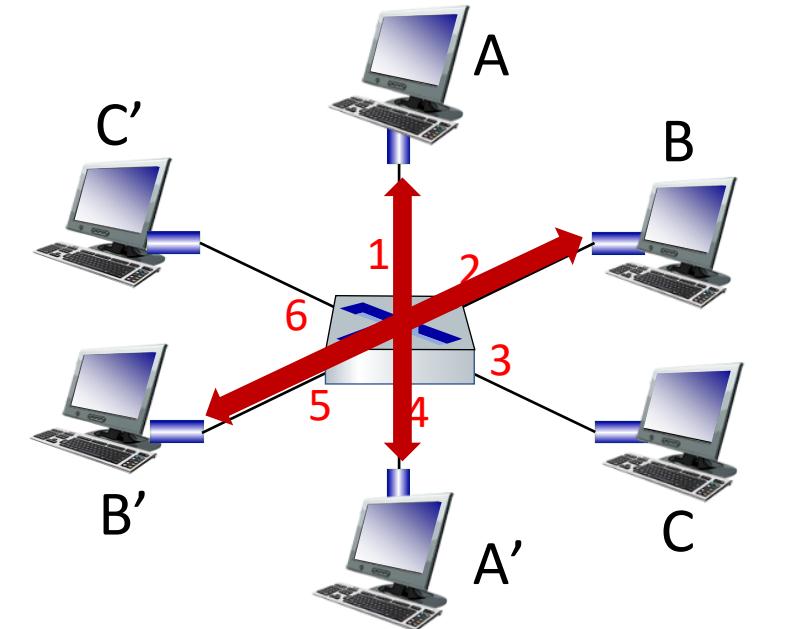
- un giorno nella vita di una richiesta web

# Switch Ethernet

- Lo switch (commutatore di pacchetti a livello di collegamento) è un dispositivo a **livello di collegamento**: ha un ruolo *attivo*
  - memorizza e inoltra (store-and-forward) frame Ethernet (o di altro tipo)
  - esamina l'indirizzo MAC di destinazione del frame in arrivo, inoltra *selettivamente* il frame in uno o più collegamenti di uscita quando il frame deve essere inoltrato in un segmento, usa CSMA/CD per accedere al segmento
- **trasparente**: gli host sono *inconsapevoli* della presenza degli switch (le cui interfacce di interconnessione agli host e router non hanno indirizzi MAC associati, o comunque non sono usati per la funzione di commutazione)
- **collegamenti eterogenei**: i collegamenti possono operare a velocità diverse e usare mezzi trasmissivi diversi; utile per evolvere la rete in maniera incrementale
- **plug-and-play, autoapprendimento**
  - non è necessario configurare gli switch

# Switch: molteplici trasmissioni simultanee

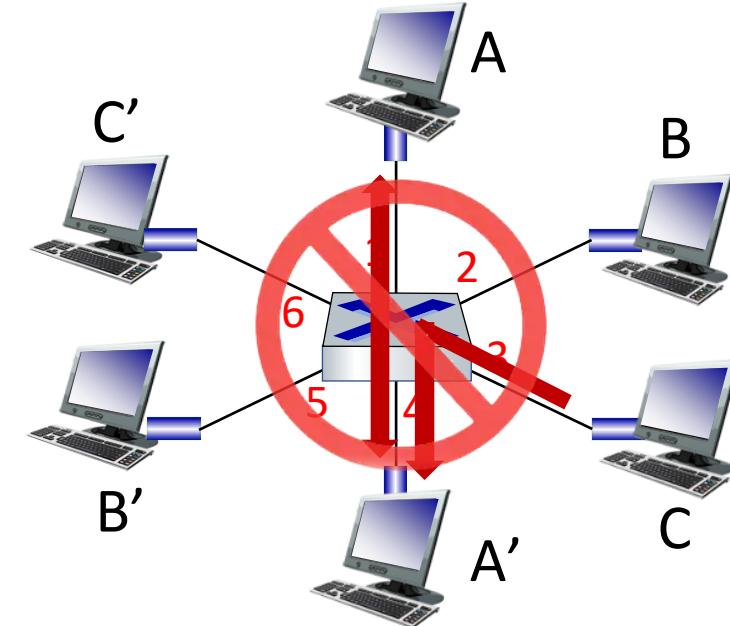
- gli host hanno connessioni dedicate, dirette con lo switch
- lo switch "bufferizza" i pacchetti
- il protocollo Ethernet è utilizzato su *ciascun* collegamento, così:
  - full-duplex: una singola coppia di nodi alle estremità del collegamento che possono trasmettere simultaneamente senza collisioni (es. perché i segnali viaggiano su fili dedicati nel cavo Ethernet), no CSMA/CD
  - half-duplex: il singolo collegamento half duplex è un dominio di collisione a sé
- **switching**: A-to-A' e B-to-B' possono trasmettere simultaneamente senza collisioni



switch con sei interfacce (1,2,3,4,5,6)

# Switch: molteplici trasmissioni simultanee

- gli host hanno connessioni dedicate, dirette con lo switch
- lo switch "bufferizza" i pacchetti
- il protocollo Ethernet è utilizzato su *ciascun* collegamento, così:
  - full-duplex: una singola coppia di nodi alle estremità del collegamento che possono trasmettere simultaneamente senza collisioni (es. perché i segnali viaggiano su fili dedicati nel cavo Ethernet), no CSMA/CD
  - half-duplex: il singolo collegamento half duplex è un dominio di collisione a sé
- **switching:** A-to-A' e B-to-B' possono trasmettere simultaneamente senza collisioni
  - ma A-to-A' e C-to-A' *non* possono accadere simultaneamente



switch con sei interfacce (1,2,3,4,5,6)

# Tabella commutazione degli switch

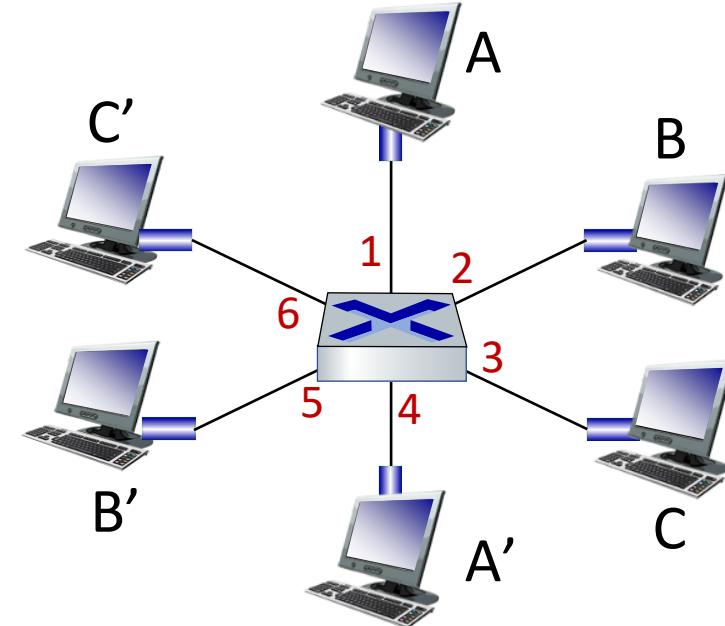
D: come sa lo switch che A' è raggiungibile tramite l'interfaccia 4, e che B' è raggiungibile dall'interfaccia 5?

R: ciascuno switch ha una **tabella di commutazione (switch table)**, ciascuna voce:

- (indirizzo MAC del nodo, interfaccia che conduce al nodo, timestamp)
- Assomiglia alle tabelle di inoltro dei router!

D: Come vengono create e mantenute le voci nella tabella di commutazione?

- qualcosa tipo un protocollo di instradamento?



# Switch: autoapprendimento

- uno switch *impara* quali nodo possono essere raggiungi attraverso quale interfaccia
  - quando un frame viene ricevuto, lo switch “impara” la posizione del mittente: segmento LAN in ingresso
  - registra la coppia mittente/posizione nella tabella di commutazione

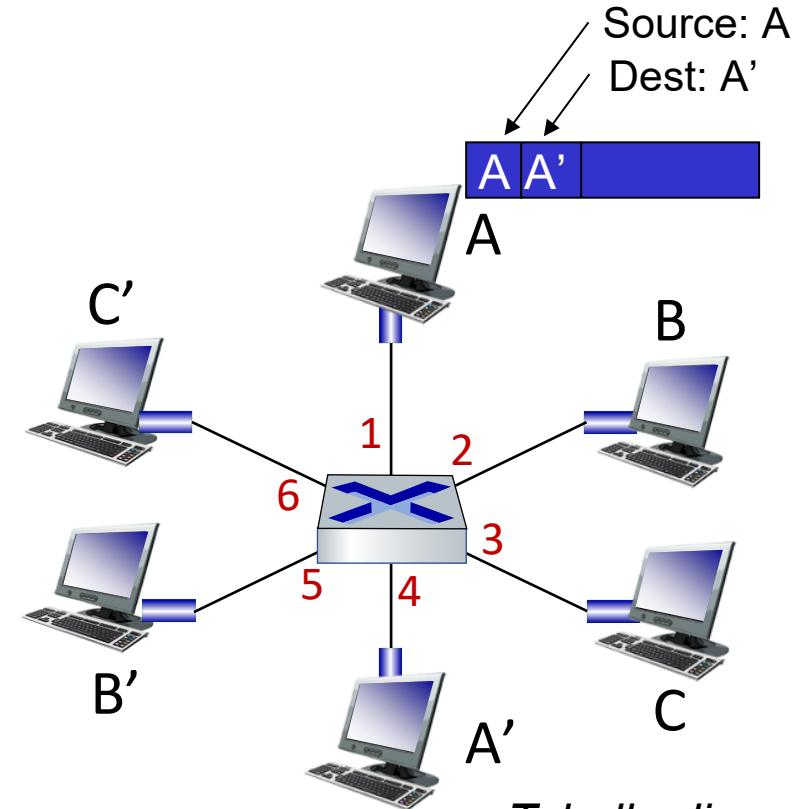


Tabella di commutazione  
(inizialmente vuota)

MAC addr	interface	TTL
A	1	60

# Switch: filtraggio e inoltro dei frame

Quando uno switch riceve un frame:

1. registra il collegamento in ingresso e l'indirizzo MAC dell'host mittente
2. indicizza la tabella degli switch utilizzando l'indirizzo MAC di destinazione
3. **se** viene trovata una voce per la destinazione  
allora {

**se** la destinazione è sul segmento dal quale è arrivato il frame

allora scarta il frame

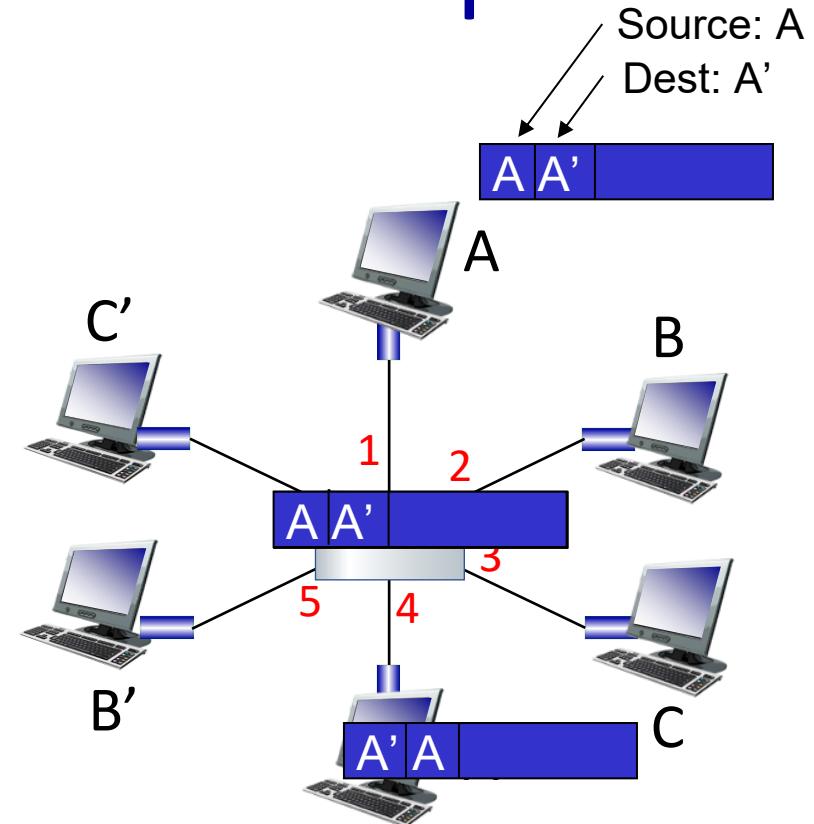
altrimenti inoltra il frame sull'interfaccia indicata dalla voce

}

altrimenti flood /\* inoltra su tutte le interfacce eccetto quella di arrivo;  
in altre parole, manda il frame in broadcast (ma non  
cambia l'indirizzo MAC di destinazione) \*/

# Autoapprendimento e inoltro: esempio

- destinazione del frame, A', posizione sconosciuta: **flood**
- posizione della destinazione A conosciuta: **invia selettivamente soltanto su un collegamento**

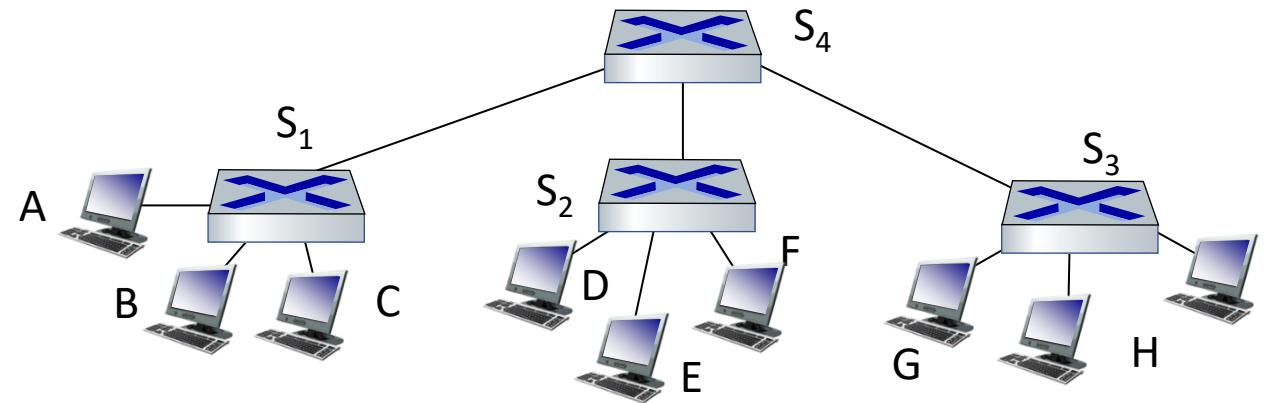


MAC addr	interface	TTL
A	1	60
A'	4	60

*tabella di commutazione /  
switch table  
(inizialmente vuota)*

# Interconnettere gli switch

gli switch con autoapprendimento possono essere interconnessi tra di loro

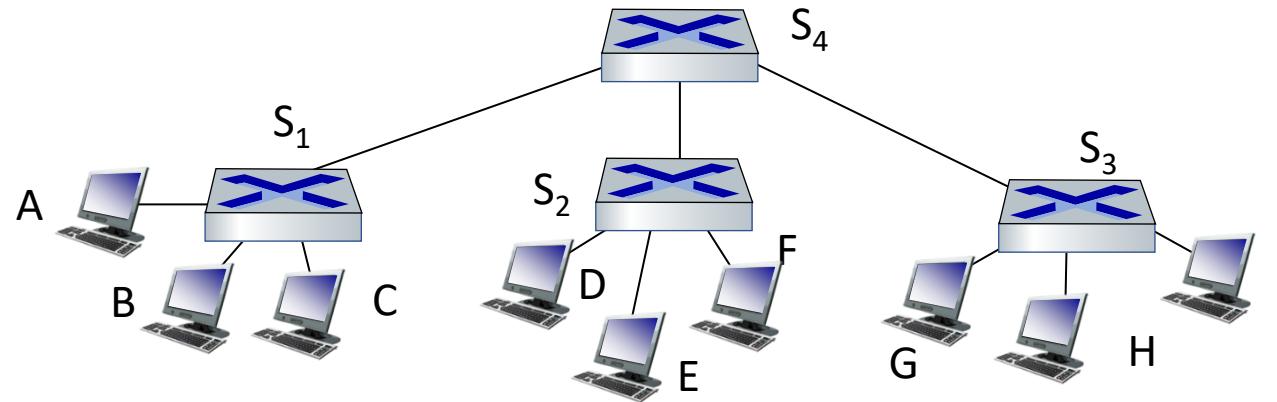


**D:** invio da A a G – come sa  $S_1$  di inoltrare il frame destinato a G attraverso  $S_4$  e  $S_3$ ?

- **R:** autoapprendimento! (funziona esattamente alla stessa maniera del caso a singolo switch!)

# Self-learning multi-switch example

Si supponga che C invii un frame a I e che I risponda a C



D: mostare le tabelle di commutazione e l'inoltro dei pacchetti in  $S_1, S_2, S_3, S_4$

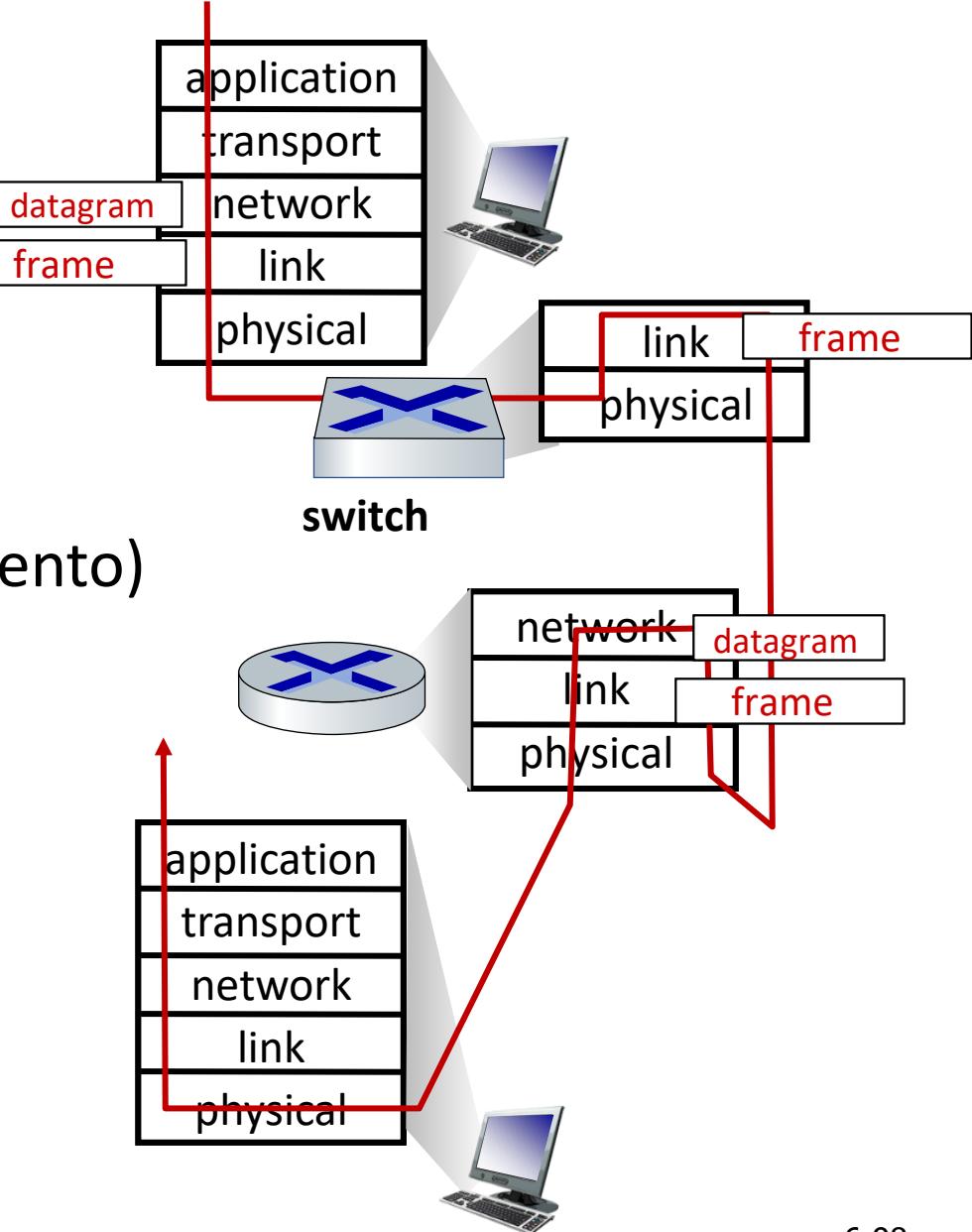
# Switch e router a confronto

Entrambi lavorano in store-and-forward:

- *router*: dispositivi a livello di rete (esaminano l'intestazione a livello di rete)
- *switch*: dispositivi a livello di collegamento (esaminano l'intestazione a livello di collegamento)

Entrambi hanno tabelle di inoltro:

- *router*: calcolano le tabelle usando algoritmi di instradamento, indirizzi IP
- *switch*: autoapprendimento della tabella di inoltro usando il flooding, indirizzi MAC



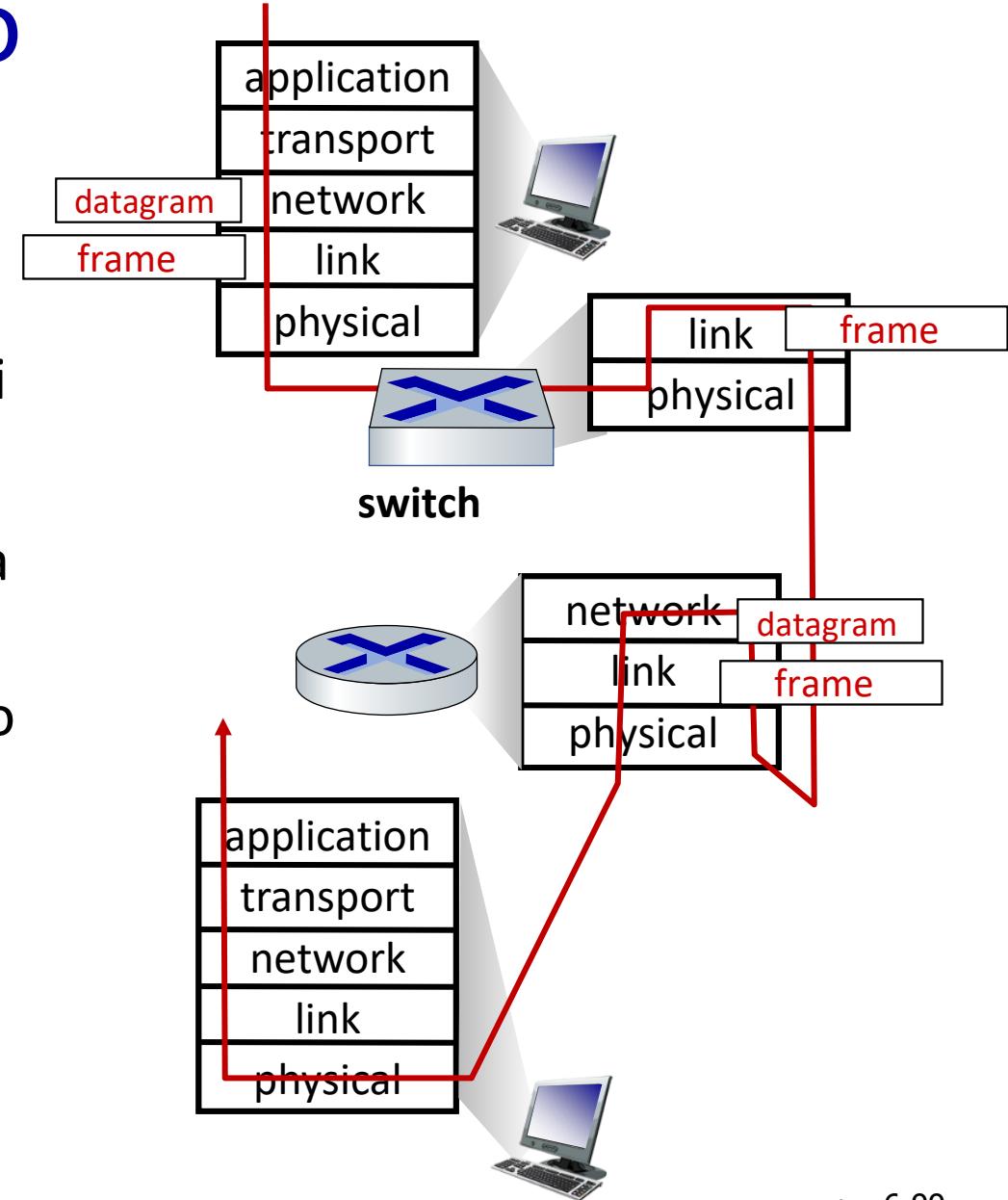
# Switch e router a confronto

## Topologia della rete:

- **router**: gli algoritmi di instradamento possono trovare percorsi ottimali (senza cicli) nonostante cicli nella topologia delle reti; inoltre, il decremento del TTL farebbe scartare i pacchetti incastrati in potenziali instradamenti ciclici (es. dovuti a errori di configurazione)
- **switch**: gli switch devono essere interconnessi a albero (anche solo logicamente, grazie al protocollo *Spanning Tree Protocol*), per evitare che il traffico broadcast (in assenza di un campo TTL nei frame) resti in circolazione potenzialmente per sempre

## Numero di nodi

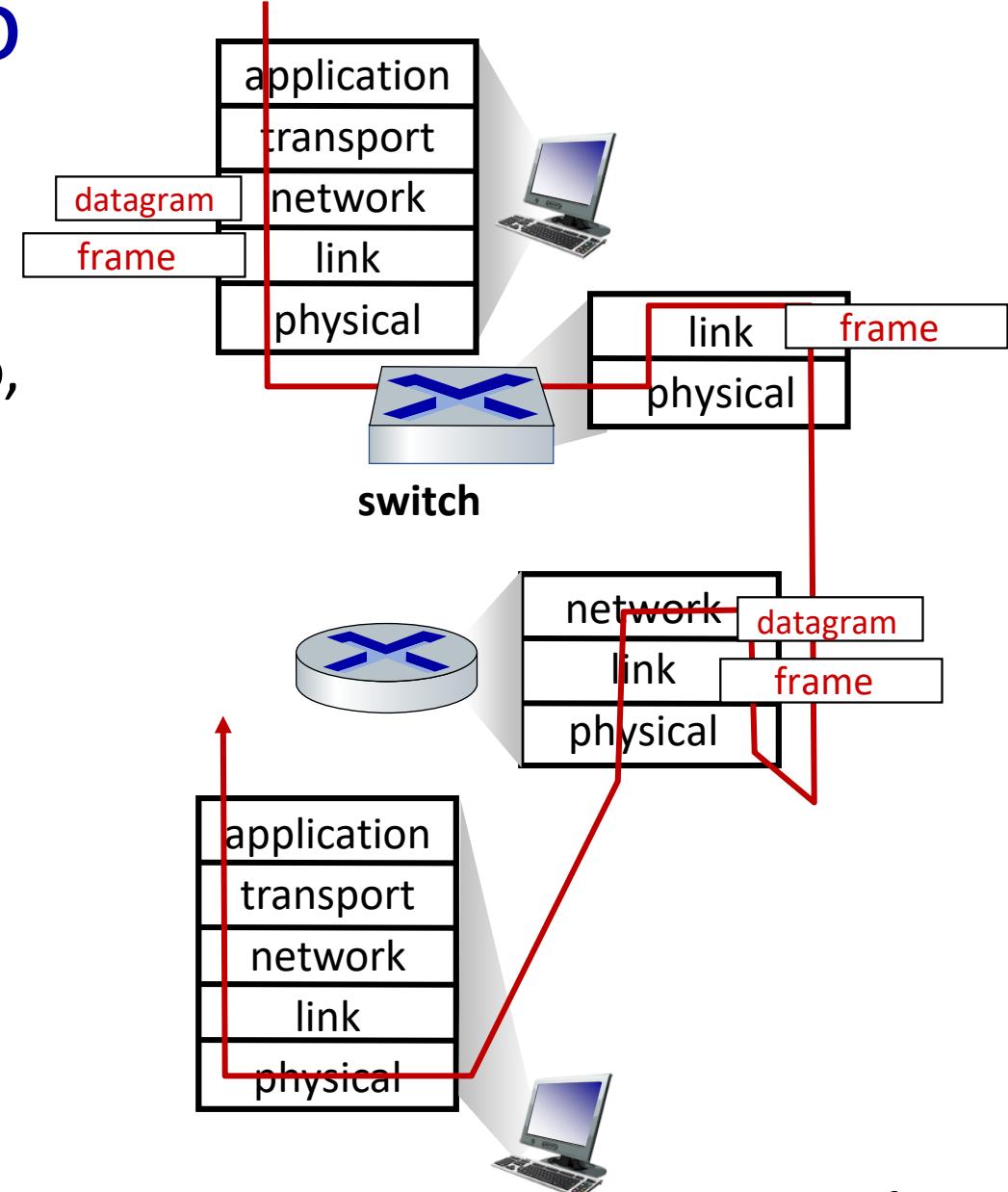
- **router**: instradamento gerarchico, aggregazione degli indirizzi, etc...
- **switch**: tabelle ARP molto grandi nei nodi, ingente traffico ARP, frame broadcast, etc...



# Switch e router a confronto

## Isolamento del traffico

- gli *switch* inviano in broadcast i frame il cui indirizzo MAC di destinazione è sconosciuto, con un effetto a valanga in presenza di molteplici switch interconnessi. I frame broadcast sono inoltrati a tutti i nodi nella rete.
- i *router* inoltrano i pacchetti in accordo a percorsi determinati dalla funzione di instradamento.



# Livello di collegamento e LAN: tabella di marcia

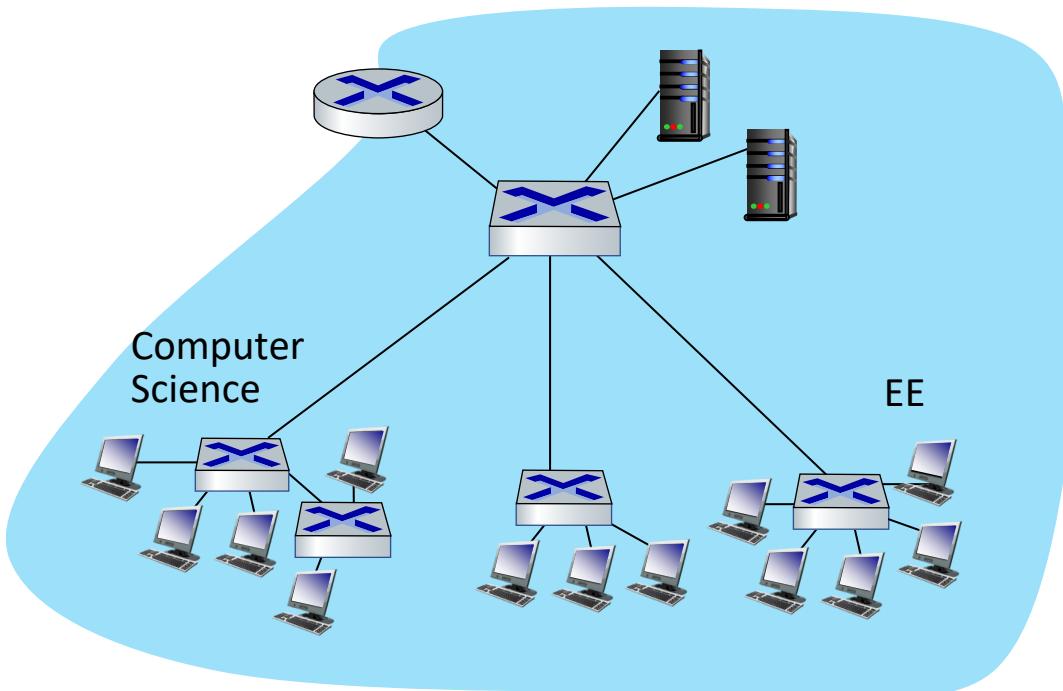
- introduzione
- rilevazione e correzione degli errori
- protocolli di acceso multiplo
- **LAN**
  - indirizzamento, ARP
  - Ethernet
  - switch
  - **VLAN**
- canali virtuali: MPLS
- Reti dei data center



- un giorno nella vita di una richiesta web

# Virtual LAN (VLAN): motivazione

D: Cosa succede quando le dimensioni della LAN aumentano e gli utenti cambiano il punto di attacco?

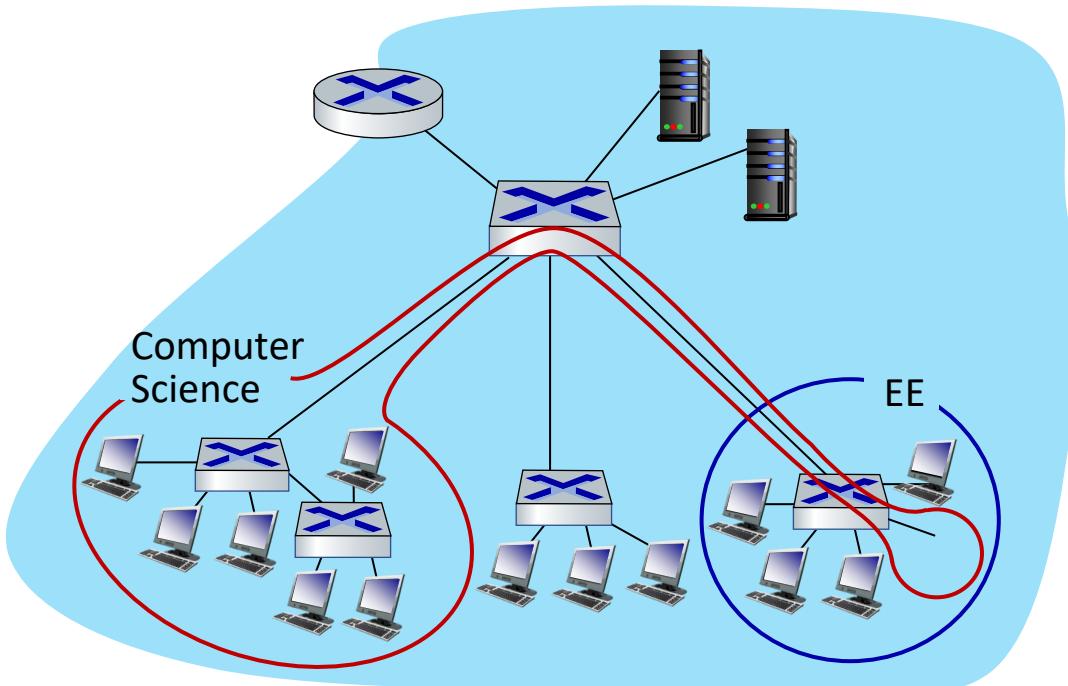


**singolo dominio di broadcast:**

- *scalabilità*: tutto il traffico broadcast di livello 2 (ARP, DHCP, MAC sconosciuto) deve attraversare l'intera LAN
- problemi di efficienza, sicurezza, privacy

# Virtual LAN (VLAN): motivazione

D: Cosa succede quando le dimensioni della LAN aumentano e gli utenti cambiano il punto di attacco?



singolo dominio di broadcast:

- *scalabilità*: tutto il traffico broadcast di livello 2 (ARP, DHCP, MAC sconosciuto) deve attraversare l'intera LAN
- problemi di efficienza, sicurezza, privacy

problemi amministrativi:

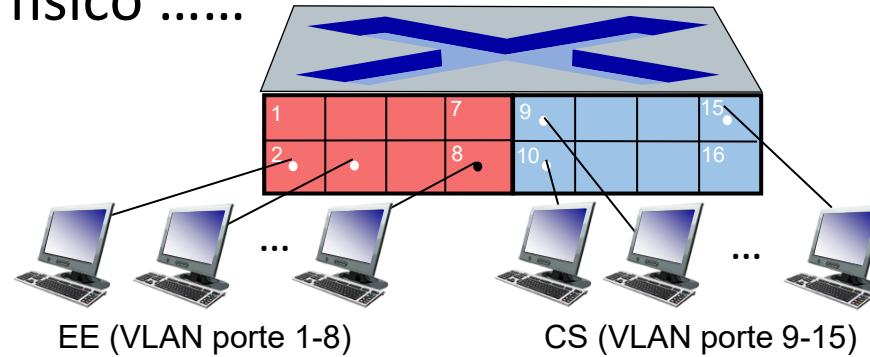
- un utente CS si sposta nell'ufficio EE - connesso *fisicamente* allo switch EE, ma vuole rimanere connesso *logicamente* allo switch CS

# VLAN basate sulle porte

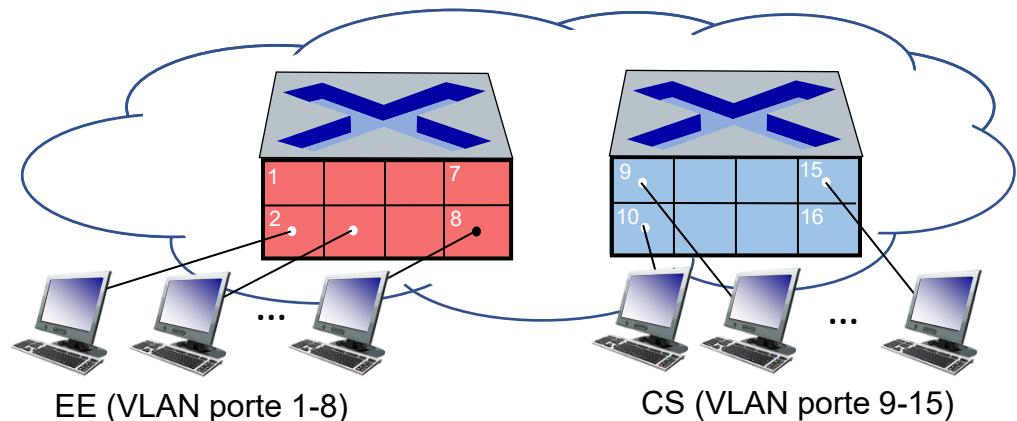
## Virtual Local Area Network (VLAN)

Gli switch che supportano le funzionalità VLAN possono essere configurati per definire più LAN *virtuali* su un'unica infrastruttura LAN fisica.

**port-based VLAN:** le porte dello switch raggruppate (tramite il software di gestione dello switch) cosicché un *singolo* switch fisico .....

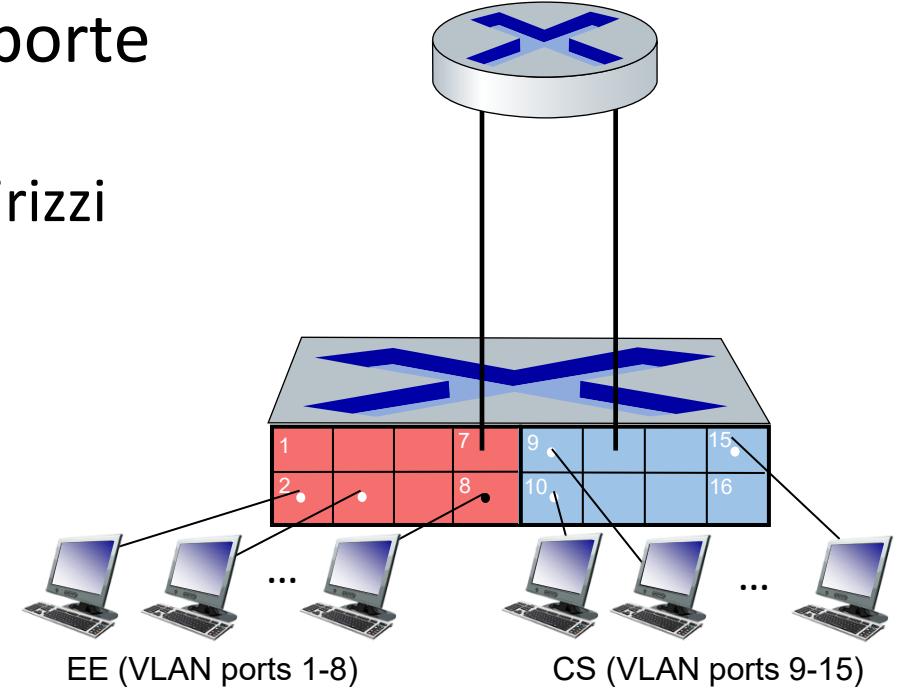


... operi come **molteplici** switch virtuali

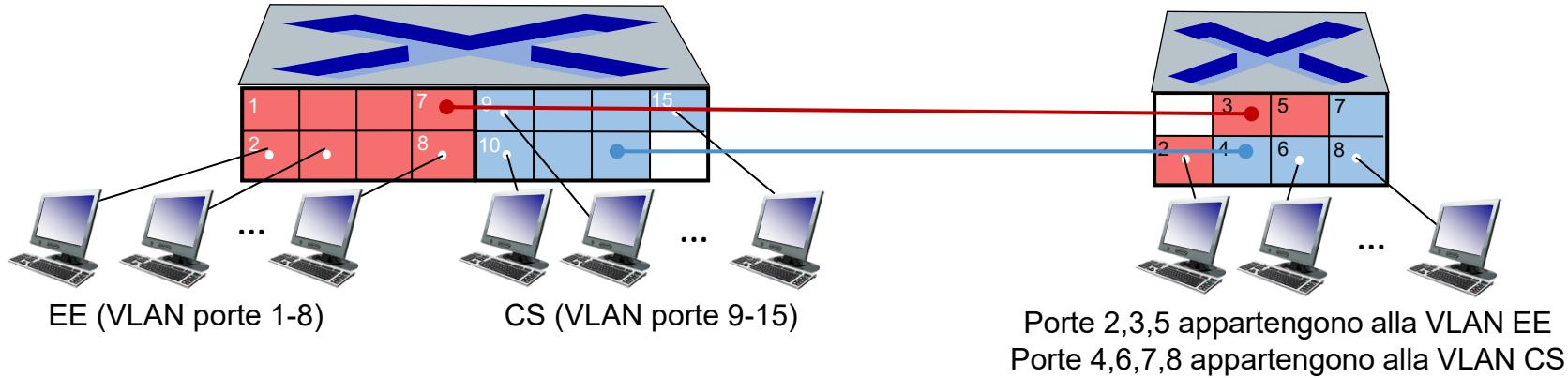


# VLAN basate sulle porte

- **isolamento del traffico:** i frame verso/da le porte 1-8 possono raggiungere *soltanto* le porte 1-8
  - Si possono define anche VLAN basate sugli indirizzi MAC degli endpoint, piuttosto che sulle porte
- **appartenenza dinamica:** le porte possono essere assegnate dinamicamente tra le VLAN
- **inoltro tra VLAN:** fatto tramite un routing (esattamente come con switch separati)
  - in pratica i produttori combinano gli switch con i router



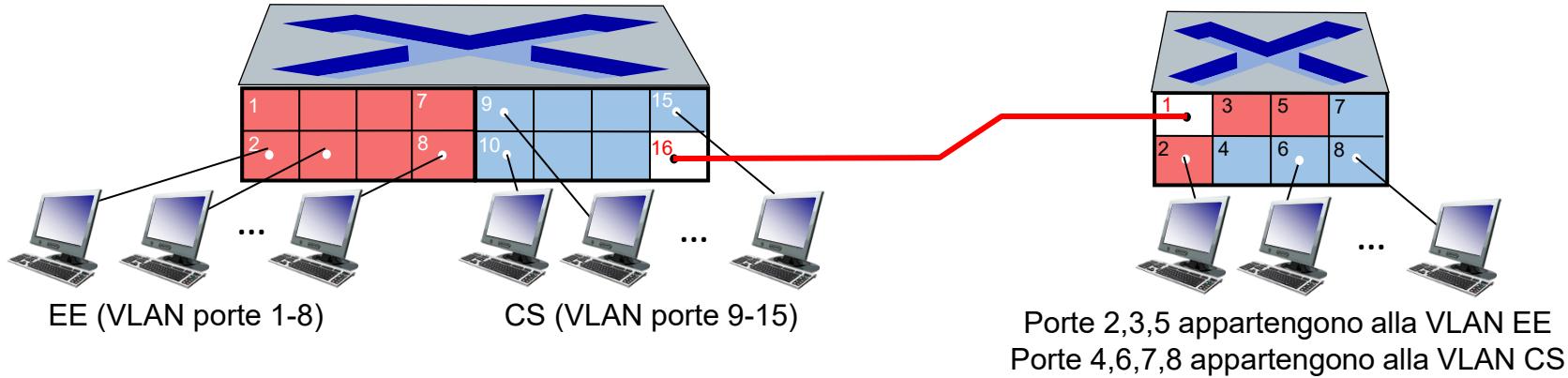
# VLAN che si estendono su più switch



**Connettere tra di loro due porte appartenenti alla stessa VLAN:**

- questa soluzione *non è scalabile*: per connettere N VLAN definite su due switch fisici, dovremmo sacrificare N porte su ciascuno switch fisico

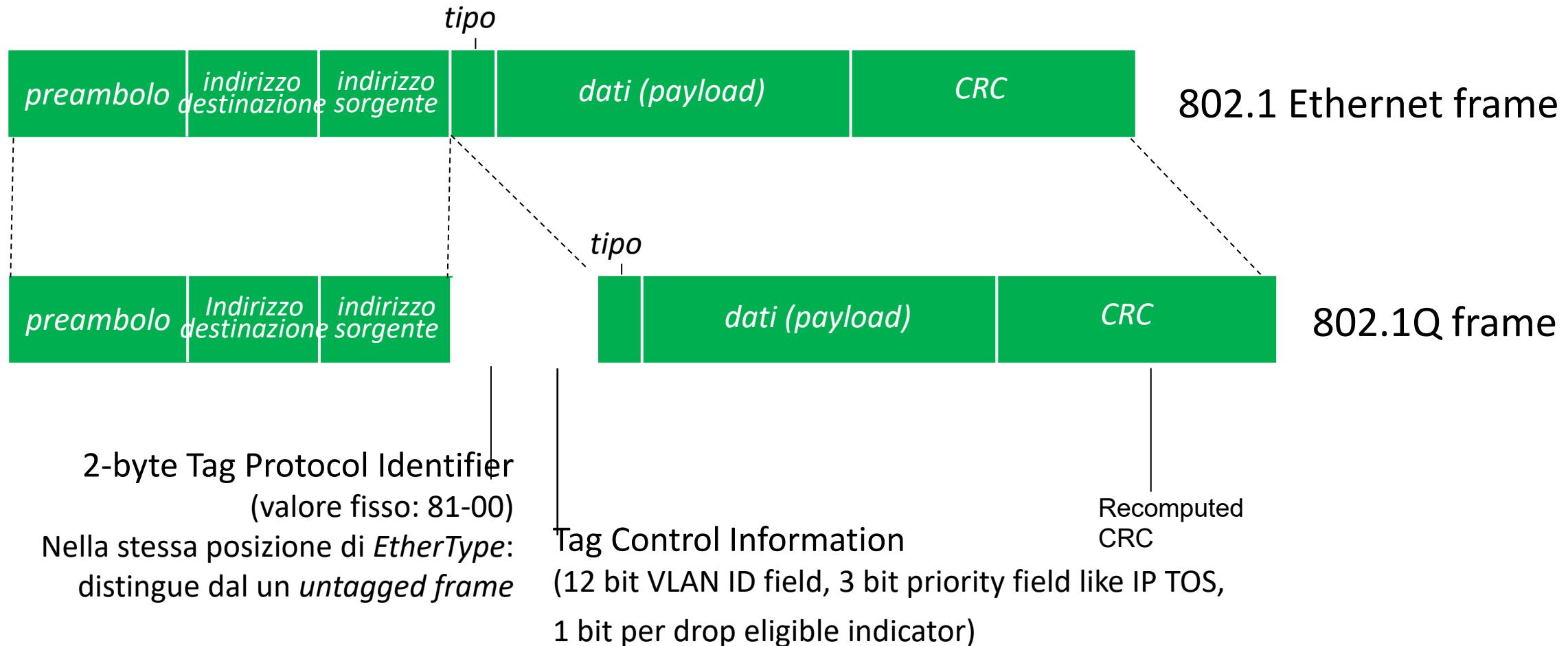
# VLAN che si estendono su più switch



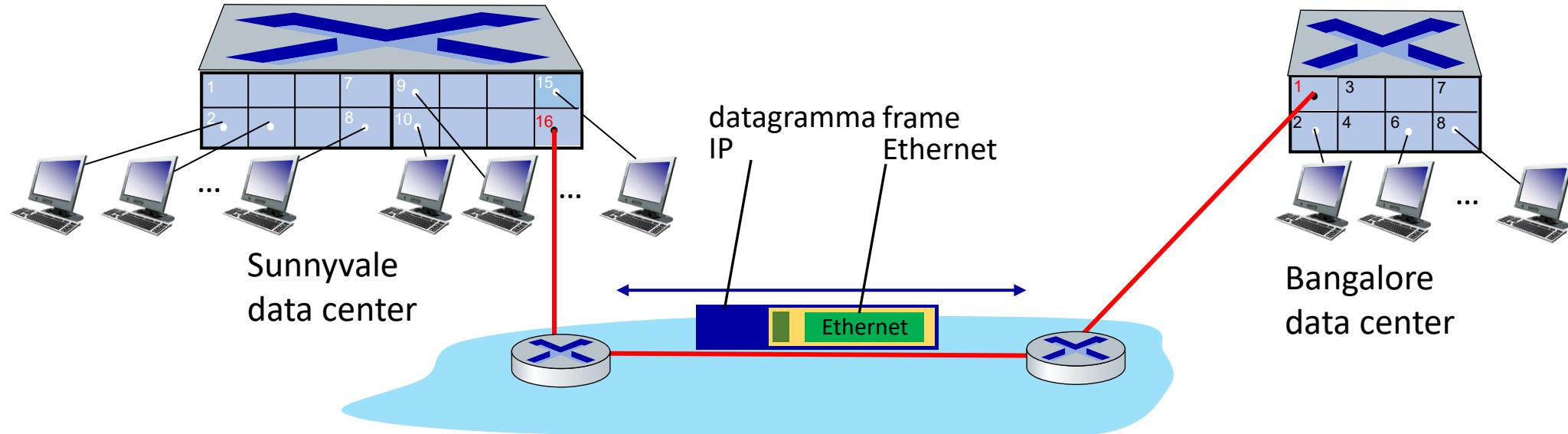
**porta trunk:** trasporta frame tra VLAN definite su più switch fisici

- i frame inoltrati all'interno della VLAN tra gli switch non possono essere frame vanilla 802.3 (devono contenere informazioni sull'ID VLAN)
- il protocollo 802.1q aggiunge/rimuove campi di intestazione aggiuntivi per i frame inoltrati tra le porte trunk

# Formato del frame VLAN 802.1Q



# EVPN: Ethernet VPN (altrimenti note come VXLAN)



Switch Ethernet di livello 2 connessi *logicamente* l'un l'altro (es., usando IP come *underlay*)

- frame Ethernet trasportati *dentro* a datagrammi IP tra siti
- “schema di *tunneling* per *sovraporre reti Layer 2 a reti Layer 3* ... funziona sull'infrastruttura di rete esistente e fornisce un mezzo per “allungare” una rete Layer 2”. [RFC 7348]