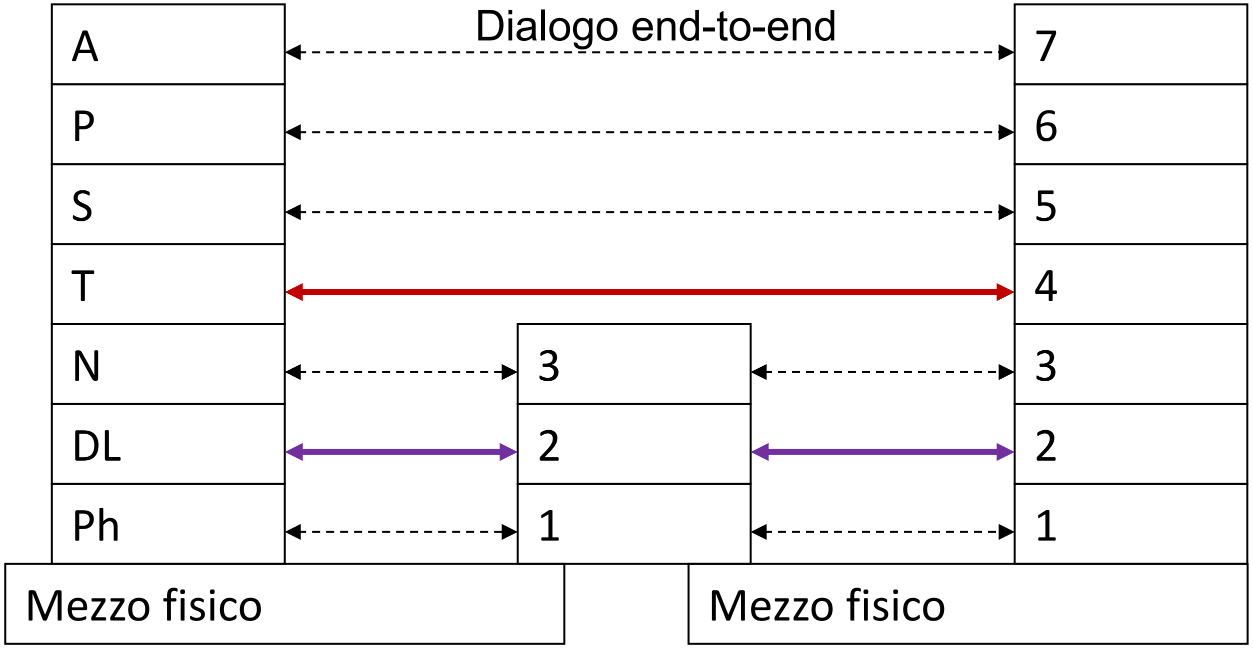
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

# Controllo del canale

Franco CALLEGATI

Dipartimento di Informatica: Scienza e Ingegneria

# Controllare il canale?



# Canale di comunicazione

- Protocolli di linea
  - Canale sequenziale a banda costante di tipo punto-punto o punto-multipunto
    - Le trame arrivano nella stessa sequenza con cui sono inviate a meno degli errori
    - Tutte sperimentano ritardi di propagazione circa uguali
- Protocolli di trasporto
  - Canale non sequenziale a capacità variabile
    - Perdite di dati (errori di trasmissione, scarto nei nodi)
    - Duplicazione dei dati
    - Ritardi variabili
    - Arrivi fuori sequenza

# Controllo del canale: strato 2

- I servizi di controllo del canale intendono
  - rendere affidabile e sicuro il servizio di collegamento che lo strato 2 offre alle entità di strato 3
- Le funzioni tipicamente svolte dallo strato 2 per il controllo del canale
  - strutturazione del flusso di dati
    - Le PDU di strato 2 sono dette trame o frame
  - controllo e gestione degli errori di trasmissione
  - controllo di flusso
  - controllo di sequenza
  - gestire il *protocollo di accesso* per un collegamento punto-multipunto
- Non tutti i protocolli di strato 2 svolgono tutte queste funzioni, alcuni implementano solo dei sottoinsiemi

# Problematiche di Sincronismo

- Nelle trasmissioni numeriche per riconoscere i bit in ricezione occorre determinare gli istanti di campionamento per ricostruire il sincronismo di cifra
- È un problema dello strato 1 che ha dei riflessi sullo strato 2
- Un circuito nel ricevitore estrae il segnale di sincronismo ma ha bisogno di agganciarsi
- Sono possibili due modalità
  - Il canale può essere tenuto sempre pieno di bit
    - L'aggancio avviene in fase di inizializzazione e viene poi sempre mantenuto
    - Il protocollo di linea deve garantire la presenza di segnale anche quando non ha dati da trasmettere
  - Il canale può avere momenti di vuoto di segnale
    - All'inizio di ogni nuova trasmissione deve essere inserito un preambolo di sincronismo

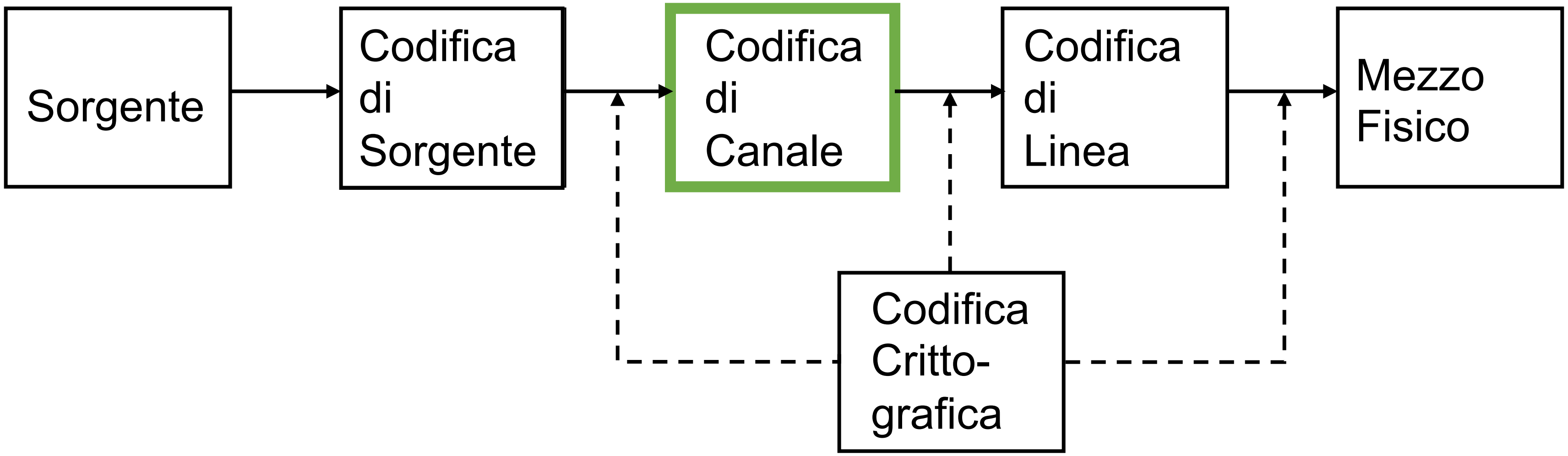
# Il sincronismo di trama

- Il sincronismo di cifra garantisce la corretta lettura dei singoli bit
- Rimane il problema di distinguere le PDU una dall’altra
- Si deve garantire il sincronismo di trama
  - Protocolli asincroni a livello di trama
    - Le trame possono iniziare e finire in ogni istante
    - Informazioni aggiuntive (nel PCI) vengono usate per riconoscere correttamente inizio e fine delle trame
  - Protocolli sincroni a livello di trama
    - Le trame devono iniziare e terminare in istanti predefiniti
    - Non sono necessarie PCI per il sincronismo

# Garantire affidabilità

- Come garantire affidabilità? Prima di consegnare i dati allo strato superiore si controllano
  - Errori di trasmissione
    - Codifica di canale con codici a rivelazione di errore
    - Conferma di ricezione e ritrasmissione
  - Sequenzialità dei dati
    - Numerazione delle unità informative
    - Conferma di ricezione e ritrasmissione
  - Flusso dei dati
    - Finestra scorrevole
    - Conferma dei dati

# La codifica di canale



Sorgente

Codifica di Sorgente

Codifica di Canale

Codifica di Linea

Mezzo Fisico

Codifica Critto- grafica

- A ognuno di questi blocchi corrisponde una Decodifica in ricezione
- Le operazioni di codifica possono essere combinate in vari modi (canale/linea, sorgente/canale, …)
- La crittografia può essere inserita in diversi punti e in diversi strati dell’architettura OSI

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

# Controllo dell’errore

# Alcune definizioni

- Codici a blocco
  - Si applica la codifica a blocchi di k bit di informazione
    - Vengono calcolati r bit di ridondanza come *funzione combinatoria* dei suddetti k bit e trasmessi n=k+r bit
- Codici convoluzionali
  - Vengono calcolati r bit di ridondanza ogni k di informazione mediante reti logiche sequenziali
    - Nel calcolo dei bit di ridondanza si tiene conto dei k bit di informazione e di *variabili di stato* dipendenti dalle operazioni passate
- Faremo riferimento d’ora in poi solamente a *codici a blocco*

# Gestione dell’errore : la codifica

- Nella codifica a blocchi
  - *k* bit vengono codificati in una parola di *n* bit aggiungendo *r=n-k* bit
  - Sono disponibili $2^n$ parole di codice per trasportare $2^k$ messaggi
    - $2^k$ sono parole di codice ammesse (valide)
    - $2^n - 2^k$ sono parole di codice non ammesse (invalide)
- Codici a rivelazione di errore
  - La ricezione di una parola di codice invalida indica la presenza di errori di trasmissione
  - Non si può dire quali siano i bit errati
  - Per garantire la trasparenza semantica è necessaria la ritrasmissione dei dati errati
- Codici a correzione di errore
  - Una parola di codice invalida
    - Indica la presenza di errori di trasmissione
    - Permette di individuare l parola di codice valida corrispondente (identifica gli errori)
    - Garantisce la trasparenza semantica in tutti i casi in cui l’ errore è correggibile

# Codifica di canale: correzione o rivelazione?

- PROBLEMA
  - Trasmissione di 1 Mbit di dati in trame lunghe 1000 bit
    - Codice a correzione di errore richiede 10 bit aggiuntivi per trama
    - Codice a rivelazione richiede 1 solo bit per trama
      - Alla rivelazione dell'errore fa seguito la ritrasmissione
- Caso 1: tasso di errore per bit del canale pari a $10^{-6}$
  - In media un errore ogni 1000 trame: bit aggiuntivi
    - 10000 bit nel caso a correzione,
    - 1000+1001=2001 nel caso a rivelazione

Conviene la rivelazione

- Caso 2: tasso di errore per bit del canale pari a $10^{-5}$
  - In media un errore ogni 100 trame: bit aggiuntivi
    - 10000 bit nel caso a correzione,
    - 1000+10*1001=11010 nel caso a rivelazione.

Circa equivalente

- Caso 3: tasso di errore per bit del canale pari a $10^{-4}$
  - In media un errore ogni 10 trame: bit aggiuntivi
    - 10000 bit nel caso a correzione,
    - 1000+111*1001=112111 nel caso a rivelazione.

Conviene la correzione

# In generale

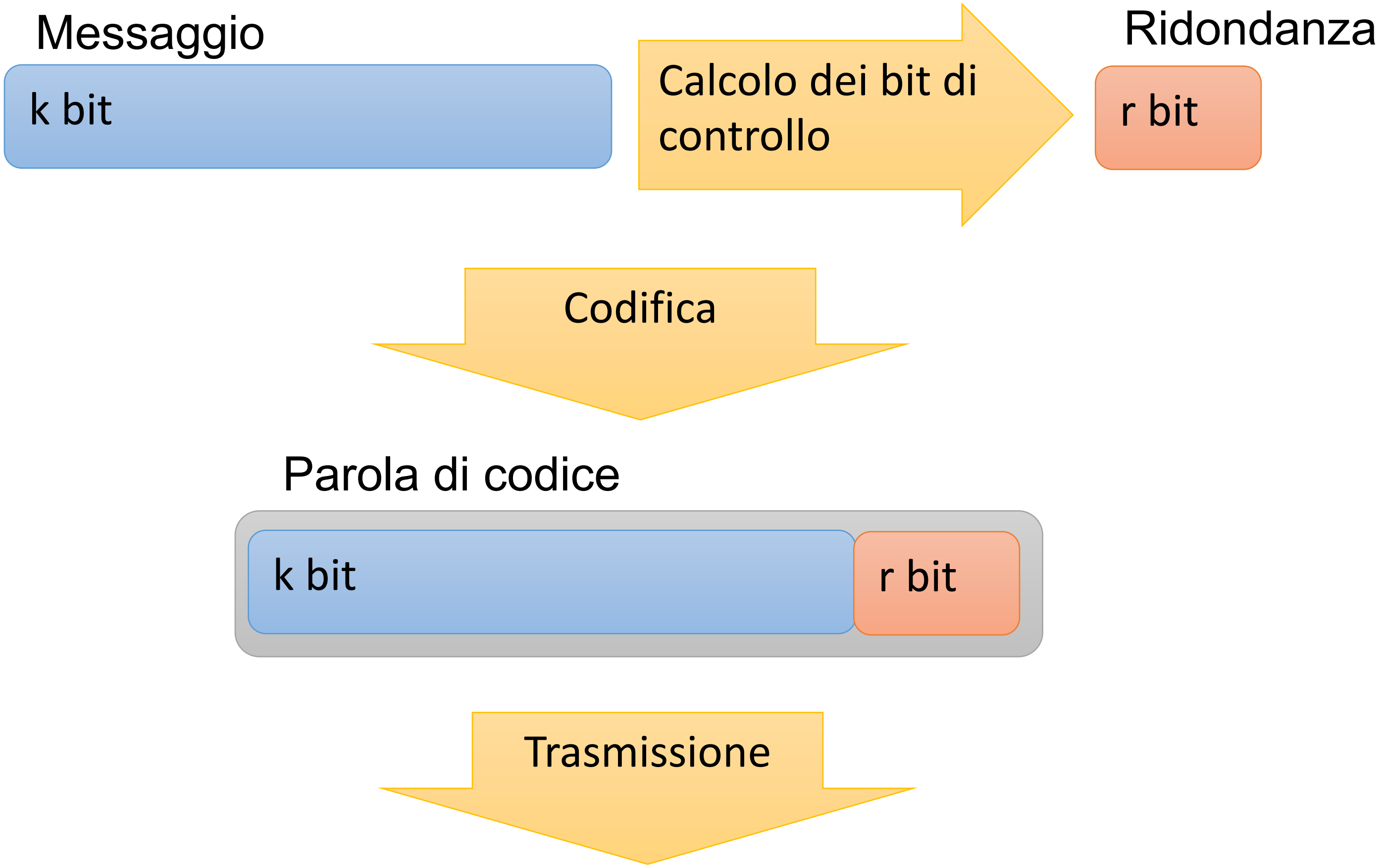
- Correzione di errore (anche forward error correction o FEC)
  - Richiede un numero abbastanza elevato di bit aggiuntivi
  - Permette la correzione dei dati errati in base ai soli dati ricevuti
- Rivelazione d'errore
  - Richiede un numero limitato di bit aggiuntivi
  - Rende necessaria la *ritrasmissione* dei dati errati
- In linea con l'esempio precedente
  - Conviene la rivelazione se il canale è affidabile per cui ci sono pochi errori
  - Conviene la correzione se il canale produce molti errori di trasmissione
- Nelle reti di solito
  - Si usano codici a correzione di errore nello strato fisico
  - Si usa la rivelazione di errore nei protocolli di linea e di trasporto

# Definizioni

- Codici lineari
  - Dati due messaggi di *k* bit $m_1$ e $m_2$
  - Ricavate le parole di codice $c_1$ e $c_2$
  - Il codice si dice lineare se $m_3 = m_1 + m_2$ da origine a $c_3 = c_1 + c_2$
- Codificatori sistematici
  - nella sequenza di *n* bit da trasmettere i *k* bit di informazione, mantenuti distinti dagli *r* bit di ridondanza, vengono trasmessi inalterati

# Uso del codice: in trasmissione

- Rivelazione d’errore
  - Codice a blocco sistematico


Messaggio

k bit

Calcolo dei bit di controllo

Ridondanza

r bit

Codifica

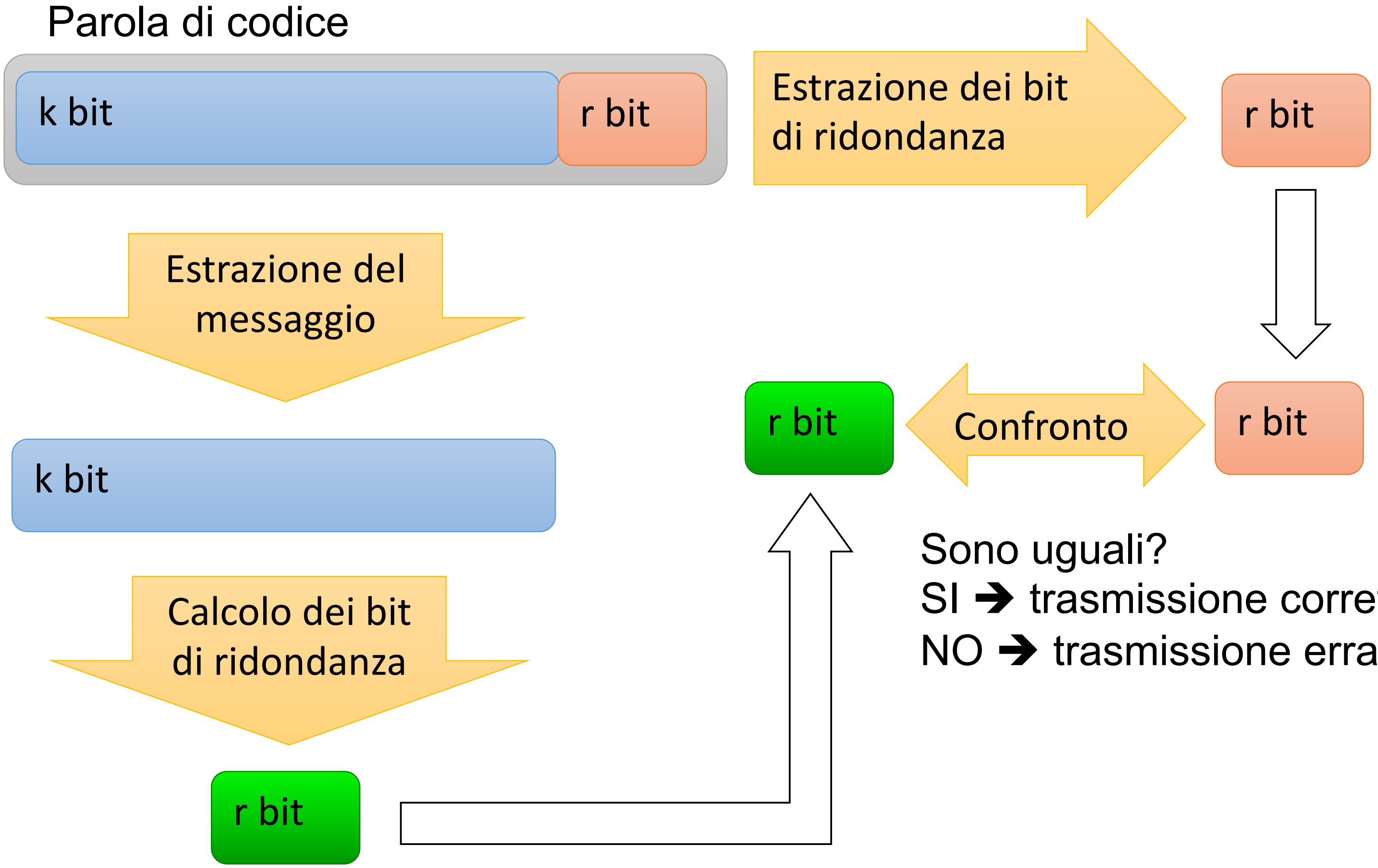
Parola di codice

k bit

r bit

Trasmissione

# Uso del codice: in ricezione



Parola di codice

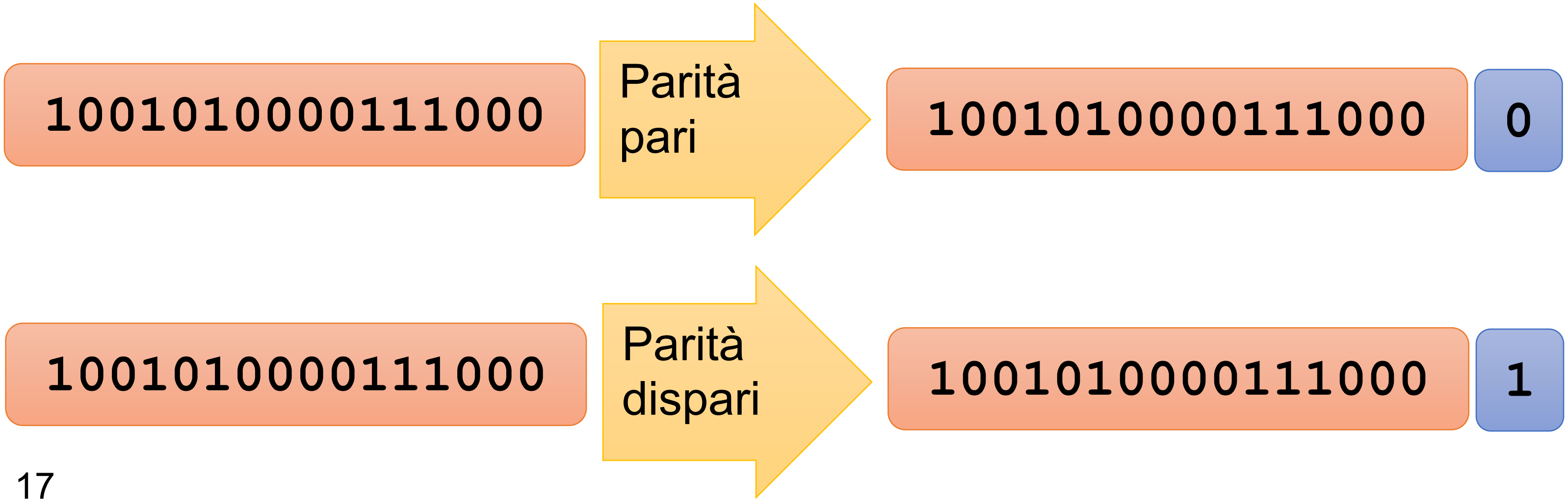
Sono uguali?
SI ➔ trasmissione corretta
NO ➔ trasmissione errata

# Il bit di parità

- Dati k bit di informazione $b_0, b_1, \dots, b_{k-1}$

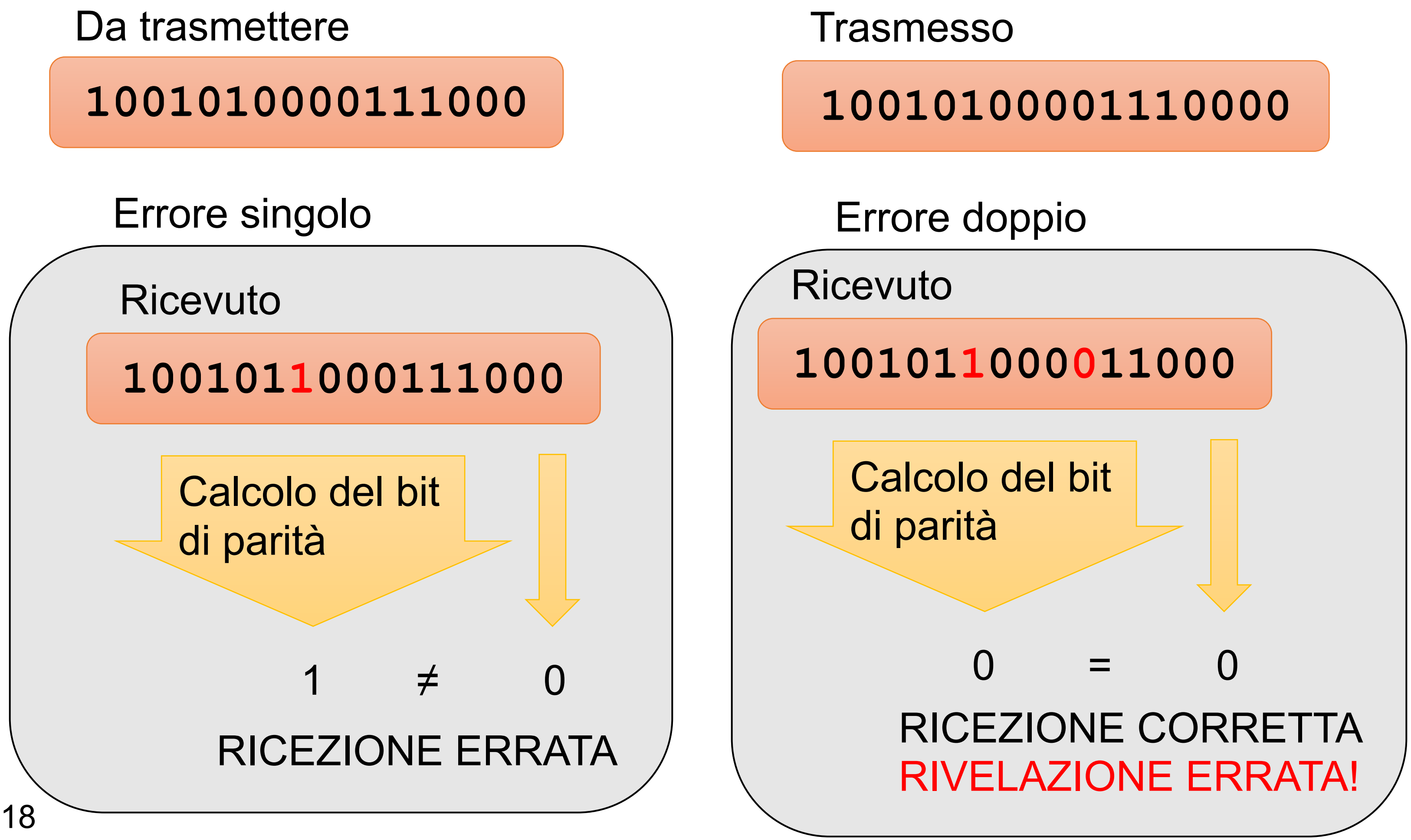
$$b_k = b_0 \oplus b_1 \oplus b_2 \oplus \dots \oplus b_{k-1} : \text{parità pari}$$
$$b_k = \text{NOT } [b_0 \oplus b_1 \oplus \dots \oplus b_{k-1}] : \text{parità dispari}$$

- Dove $\oplus$ è l'operazione di OR esclusivo
- $r = 1$ un solo bit di ridondanza per qualunque dimensione del blocco dati $k$



# Proprietà

- Rivela sempre un numero dispari di errori
- Fallisce con un numero pari di errori



Da trasmettere

`1001010000111000`

Trasmesso

`10010100001110000`

Errore singolo

Ricevuto

`10010110000111000`

Calcolo del bit di parità

1 ≠ 0

RICEZIONE ERRATA

Errore doppio

Ricevuto

`10010110000011000`

Calcolo del bit di parità

0 = 0

RICEZIONE CORRETTA

RIVELAZIONE ERRATA!

# Internet checksum

- Nei protocolli di Internet vengono solitamente utilizzati codici a blocchi sistematici
- Sono estensioni del bit di parità, volte ad estenderne le prestazioni
- Si applica su parole di 16 bit, indipendente dalla lunghezza complessiva del blocco dati

RFC1071 - 1988

```
In outline, the Internet checksum algorithm is very
simple:
(1)Adjacent octets to be checksummed are paired to form
16-bit integers, and the 1's complement sum of these 16-
bit integers is formed.
(2) To generate a checksum, the checksum field itself is
cleared,  the 16-bit 1's complement sum is computed over
the octets concerned, and the 1's complement of this sum
is placed in the checksum field.
(3)To check a checksum, the 1's complement sum is
computed over the same set of octets, including the
checksum field.  If the result is all 1 bits (-0 in 1's
complement arithmetic), the check succeeds.
```

# Somma complemento a 1

- La somma complemento a 1 è simile al calcolo binario intero senza segno (somma complemento a 2) ma differisce per l’uso dei riporti
- Se una somma genera un riporto questo viene aggiunto al risultato

```
Somma complemento a 1
 11110010 +
 11110100
---------
111100110
        1
---------
 11100111
```

# Proprietà

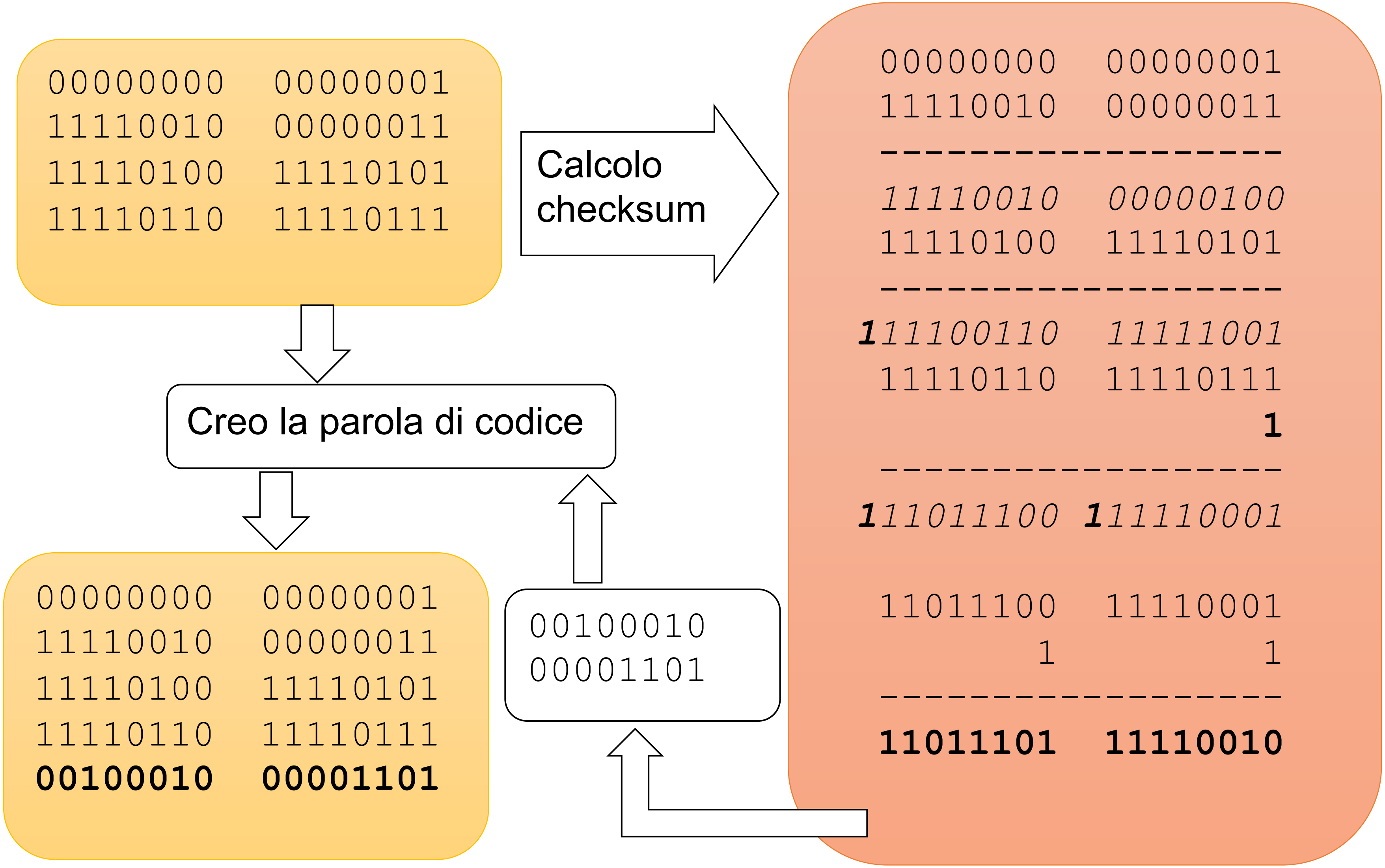
- Blocco dati fatto di byte A, B, C, D, E, F, G, …
- Parole di 16 bit [A,B], [C,D], [E,F], [G,H]

- Proprietà commutativa e associativa
  - [A,B]+[C,D] = [C,D]+[A,B]
  - ([A,B]+[C,D])+[E,F] = [A,B]+([C,D]+[E,F])

- Indipendenza dall’ordine dei byte
  - [A,B]+[C,D] = [X,Y] allora [B,A]+[D,C] = [Y,X]
    - Questa proprietà è molto importante perché rende il calcolo indipendente dalla rappresentazione del numero a livello di sistema hardware "big-endian" o "little-endian"

# Esempio

- Devo calcolare il checksum di 64 bit, raggruppabili in 4 parole da 16 bit



00000000 00000001
11110010 00000011
11110100 11110101
11110110 11110111

Calcolo checksum

```
 00000000  00000001
 11110010  00000011
 -------------------
 11110010  00000100
 11110100  11110101
 -------------------
111100110  11111001
 11110110  11110111
                  1
 -------------------
111011100 111110001

 11011100  11110001
        1         1
 -------------------
 11011101  11110010
```

00100010
00001101

Creo la parola di codice

00000000 00000001
11110010 00000011
11110100 11110101
11110110 11110111
**00100010 00001101**

# In ricezione

00000000 00000001
11110010 00000011
11110100 11110101
11110110 11110111
**00100010 00001101**

Eseguo la somma

```
  00000000   00000001
  11110010   00000011
  --------------------
  11110010   00000100
  11110100   11110101
  --------------------
 111100110   11111001
  11110110   11110111
  --------------------
1011011100 111110000
  00100010   00001101
  --------------------
1011111110 111111101

  11111110 111111101
         1        10
  --------------------
  11111111   11111111
```

Checksum giusto
Ricezione corretta

# Errore

```
00000000   00000001
11110010   00000011
11110100   11110101
11110110   11101111
00100010   00001101
```

Eseguo la somma

```
   00000000   00000001
   11110010   00000011
   -------------------
   11110010   00000100
   11110100   11110101
   -------------------
  111100110   11111001
   11110110   11101111
   -------------------
 1011011100 111111000
   00100010   00001101
   -------------------
 10111111101000000101

   11111110 000000101
         10         10
   -------------------
  100000000   00000111
```

Checksum inesatto
Ricezione errata

# Algebra binaria e codici polinomiali

- L’algebra si costruisce sull’insieme dele cifre binarie 0 e 1
  - $\mathbb{A} = \{0, 1\}$
- Operazioni:
  - Or esclusivo ⊕ (somma e sottrazione)
    - a ⊕ (b ⊕ c) = (a ⊕ b) ⊕ c infatti ad esempio 1 ⊕ (1 ⊕ 0) = (1 ⊕ 1) ⊕ 0 = 0
    - Ha elemento neutro ed opposto
      - 0 elemento neutro infatti 1 ⊕ 0 = 1 e 0 ⊕ 0 = 0
      - 1 opposto infatti 1 ⊕ 1 = 0 e 0 ⊕ 1 = 1
    - Vale la proprietà commutativa
      - Infatti 1 ⊕ 0 = 0 ⊕ 1 = 1
    - Genera un gruppo abeliano
  - Moltiplicazione
    - Esiste l’elemento neutro
      - 1 infatti 0 x 1 = 0 e 1 x 1 = 1
    - Vale la proprietà commutativa infatti 1 x 0 = 0 x 1 = 0
    - Vale la proprietà distributiva infatti 1 x (1 ⊕ 0) = 1 x 1 ⊕ 1 x 0) = 1
  - Si genera un anello
- Ne risulta un’algebra analoga a quella ordinaria ma limitata alle cifre binarie
- L'insieme polinomi con variabile x e coefficienti in un anello formano un anello ed ereditano le operazioni e le loro proprietà

# Codici polinomiali

- Basati sull'uso di polinomi in un'algebra binaria
  - *k* bit vengono posti in corrispondenza con un polinomio di grado *k-1* nella variabile binaria *x*:

$$P_{k-1}(x) = b_0 + b_1x + b_2x^2 + ... + b_{k-1}x^{k-1}$$

- Vengono calcolati i bit di ridondanza utilizzando operazioni sui polinomi
- Polinomio generatore
  - Viene stabilito un polinomio di grado *r* noto a trasmettitore e ricevitore
  - $G_r(x)$ determina le proprietà di rivelazione del codice

# Polinomio trasmesso

- Per calcolare polinomio $T_{n-1}(x)$ da trasmettere:
  - Si moltiplica il polinomio $P_{k-1}(x)$ per $x^r$
    - $r$ bit a zero posti in coda
  - Si esegue la divisione polinomiale fra $P_{k-1}(x)\, x^r$ e $G_r(x)$ ottenendo un quoziente ed un resto

$$P_{k-1}(x)\, x^r = G_r(x)\, Q_{k-1}(x) \oplus R_{r-1}(x)$$

- Notando che nell'algebra adottata somma e sottrazione coincidono, si trasmette

$$T_{n-1}(x) = P_{k-1}(x)\, x^r \oplus R_{r-1}(x) = G_r(x)\, Q_{k-1}(x)$$

- Proprietà di $T_{n-1}(x)$
  - Realizza una codifica di tipo sistematico perché i bit di resto, al più $r$ bit, vanno a sovrapporsi agli $r$ zeri in coda
  - È multiplo di $G_r(x)$

# Polinomio ricevuto

- Il ricevitore riceve una sequenza di *n* bit che corrisponde al polinomio ricevuto

$$T'_{n-1}(x)$$

- Se si verifica un errore di trasmissione

$$T'_{n-1}(x) \neq T_{n-1}(x)$$

- Esisterà un polinomio E(x) tale che

$$T'_{n-1}(x) = T_{n-1}(x) + E(x)$$

- E(x)
  - ha coefficienti non nulli in corrispondenza dei bit in cui $T'_{n-1}(x)$ differisce da $T_{n-1}(x)$
  - rappresenta in forma polinomiale gli eventuali errori e per questo si dice *polinomio errore*

# Rivelazione dell'errore

- Il ricevitore esegue la divisione:

$$T'_{n-1}(x)/G_r(x) = (T_{n-1}(x) + E(x))/G_r(x) = T_{n-1}(x)/\ G_r(x)\ \ + E(x)/\ G_r(x)$$

- Il primo di questi termini ha sempre resto 0

- Se $E(x) \neq 0$ e $E(x)$ non è divisibile per $G_r(x)$, allora il resto della divisione precedente risulta diverso da 0 e viene quindi rilevato l'errore

- Per rilevare gli errori si deve quindi evitare che:

$$Resto[E(x)/G_r(x)]\ \ = 0$$

- $G_r(x)$ va scelto per minimizzare la probabilità di non rivelare un errore
  - affinché due polinomi siano divisibili è comunque necessario che il grado del numeratore sia maggiore o uguale al grado del denominatore

# Capacità del codice e scelta di $G_r(x)$

- Un singolo errore
  - $E(x) = x^i$
    - è sufficiente che in $G_r(x)$ vi siano almeno due bit a 1
- Numero dispari di errori
  - Se $G_r(x)$ è multiplo di $(1+x)$ non divide mai un polinomio con numero dispari di termini
  - Se si sceglie $G_r(x) = 1+x$ , il codice polinomiale fornisce 1 singolo bit di ridondanza eguale al bit di parità
- 2 errori
  - $E(x) = x^i+x^j = x^j(x^h+1)$. Esistono diversi polinomi che non dividono mai $(x^h+1)$. ITU ha proposto il seguente polinomio :

$$G_{16}(x) = x^{16} + x^{12} + x^5 +1$$

# Errori a burst

- Nelle reti di telecomunicazione è frequente una distribuzione non uniforme degli errori, con concentrazione degli stessi in certi intervalli
  - filotto (burst) di bit lungo *k* cui bit intermedi sono inaffidabili (supponiamo abbiano una probabilità di essere errati pari al 50%)
  - rappresentato da un polinomio di grado $k - 1$
- Si possono avere i seguenti casi:
  - $k - 1 < r$ : l'errore viene sempre rilevato;
  - $k - 1 = r$ : si ha resto nullo se $E(x) = G_r(x)$
    - questo evento può verificarsi con probabilità = $1/2^{r-1}$
  - $k - 1 > r$ : il resto ha valore casuale e l'errore sfugge se il resto è nullo (r bit a 0)
    - questo evento può con probabilità = $1/2^r$

# Esercizo 1- (a)

- Si vuole trasmettere da S a D la seguente stringa di bit utilizzando un codice polinomiale per verificare che la trasmissione avvenga senza errori.
- Informazione Iniziale: 101101000101
- Il polinomio di grado k-1, $P_{k-1}(x)$ nella variabile binaria x è il seguente:

$$P(x) = x^{11}+0+x^9+x^8+0+x^6+0+0+0+x^2+0+1$$

- Il polinomio generatore è il seguente:

$$G(x)=x^2+x+1$$

Quindi r = 2

# Esercizio 1- (b)

- Determiniamo ora gli n bit realmente trasmessi e verifichiamo la resistenza agli errori di trasmissione che si ottiene attraverso il metodo dei codici polinomiali.
- G(x) è un polinomio di secondo grado per cui occorre moltiplicare il polinomio P(x) per $x^2$:

$$P(x)x^2= x^{13}+0+x^{11}+x^{10}+0+x^8+0+0+0+x^4+0+ x^2+0+0$$

- Si devono calcolare gli n bit che verranno trasmessi

# Esercizio 1- (c)

- Si divide P(x) $x^2$ per G(x), per ottenere $T_{n-1}(x)$:

$x^{13}+0+x^{11}+x^{10}+0+x^{8}+0+0+0+x^{4}+0+x^{2}+0+0$

$x^{13}+x^{12}+x^{11}$

$/ \quad x^{12}+ 0+x^{10}+0+x^{8}+0+0+0+x^{4}+0+x^{2}+0+0$

$x^{12}+x^{11}+x^{10}$

$/ \quad x^{11}+0 +0+x^{8}+0+0+0+x^{4}+0+x^{2}+0+0$

$x^{11}+x^{10}+x^{9}$

$/ \quad x^{10}+x^{9}+x^{8}+0+0+0+x^{4}+0+x^{2}+0+0$

$x^{10}+x^{9}+x^{8}$

$/ \quad / \quad / \qquad x^{4}+0+x^{2}+0+0$

$x^{4}+x^{3}+x^{2}$

$/ \quad x^{3}+0+0+0$

$x^{3}+x^{2}+x$

$/ \quad x^{2}+x+0$

$x^{2}+x+1$

$/ \quad / \quad 1$

$x^2+x+1$

$x^{11} + x^{10} + x^{9} + x^{8} + x^{2} + x + 1$

# Esercizio 1- (d)

- Dalla divisione si ottiene:
  - $R(x) = 1$
  - $Q(x) = x^{11} + x^{10} + x^9 + x^8 + x^2 + x + 1$

- quindi il polinomio da trasmettere è
  - $Tn\text{-}1(x) = x^{13}+0+x^{11}+x^{10}+0+x^8+0+0+0+x^4+0+x^2+0+1$

- In conclusione la sequenza di bit da trasmettere è dunque:

10110100010101

# Esercizio 1- (e)

- Il polinomio ricevuto alla destinazione è dato da:
  - $T^1_{n-1}(x)= T_{n-1}(x)+E(x)$;
  - $E(x)$ rappresenta il polinomio errore.
  - $E(x)$ ha coefficienti diversi da 0 in corrispondenza dei bit $T_{n-1}(x)$ che vengono corrotti dall'errore.
- Il ricevitore verifica la correttezza dei dati ricevuti eseguendo la seguente divisione:

$$\frac{T^{'}_{n-1}(x)}{G(x)} = \frac{T_{n-1}(x)+E(x)}{G(x)}$$

# Esercizo 1- (f)

- Consideriamo i seguenti E(x)
  - $E(x)=x^9+x^8$
  - $E(x)=x^4+x^3+x^2$

- Considerando il primo polinomio E(x):
  - $T_{n-1}(x) = x^{13}+0+x^{11}+x^{10}+0+x^8+0+0+0+x^4+0+x^2+0+1$
  - $T^1_{n-1}(x)=$ **$T_{n-1}(x)+E(x)=$** $x^{13}+0+x^{11}+x^{10}+x^9+0+0+0+0+x^4+0+x^2+0+1$
  - $T^1_{n-1}(x) =$ **l101110000101l01l**

- Per verificare se l' errore viene rilevato si deve dividere $T^1_{n-1}(x)$ per $G(x) = x^2+x+1$

# Esercizo 1- (g)

- $T^1_{n-1}(x)$ diviso $G(x)$:

$x^{13}+0\ \ +x^{11}+x^{10}+x^9+0+0+0+0+x^4+0+x^2+0+1$

$x^{13}+x^{12}+x^{11}$

/ $x^{12}+ 0 +x^{10}$

$x^{12}+x^{11}+x^{10}$

/ $x^{11} + 0\ \ + x^9$

$x^{11}+x^{10} + x^9$

/ $x^{10}+0+0$

$x^{10}+x^9+x^8$

/ $x^9+x^8+0$

$x^9+x^8+x^7$

/ / $x^7+0 +0$

$x^7+x^6+x^5$

/ $x^6+x^5+x^4$

$x^6+x^5+x^4$

/ / / $0+x^2+0+1$

$x^2+x+1$

/ $+x+0$

$x^2+x+1$

$x^{11}+x^{10}+x^9+x^8+ x^7+ x^5+x^4+1$

# Esercizo 1- (h)

- Eseguiamo la divisione per il secondo caso:
  - $E(x)=x^4+x^3+x^2$

$x^{13}+0\ +x^{11}+x^{10}+0+x^8+0+0+0+0+x^3+0+0+1$ | $x^2+x+1$

Quoziente: $x^{11}+x^{10}+x^9+x^8+x+1$

$x^{13}+x^{12}+x^{11}$

/ $x^{12}+0+x^{10}$

$x^{12}+x^{11}+x^{10}$

/ $x^{11}+0\ +0$

$x^{11}+x^{10}+x^9$

/ $x^{10}+x^9+x^8$

$x^{10}+x^9+x^8$

/ / /

$x^3+0+0$

$x^3+x^2+x$

/ $x^2+x+1$

$x^2+x+1$

/ / /

- In questo caso l' errore **NON VIENE RIVELATO**

# Authomatic Repeat Request

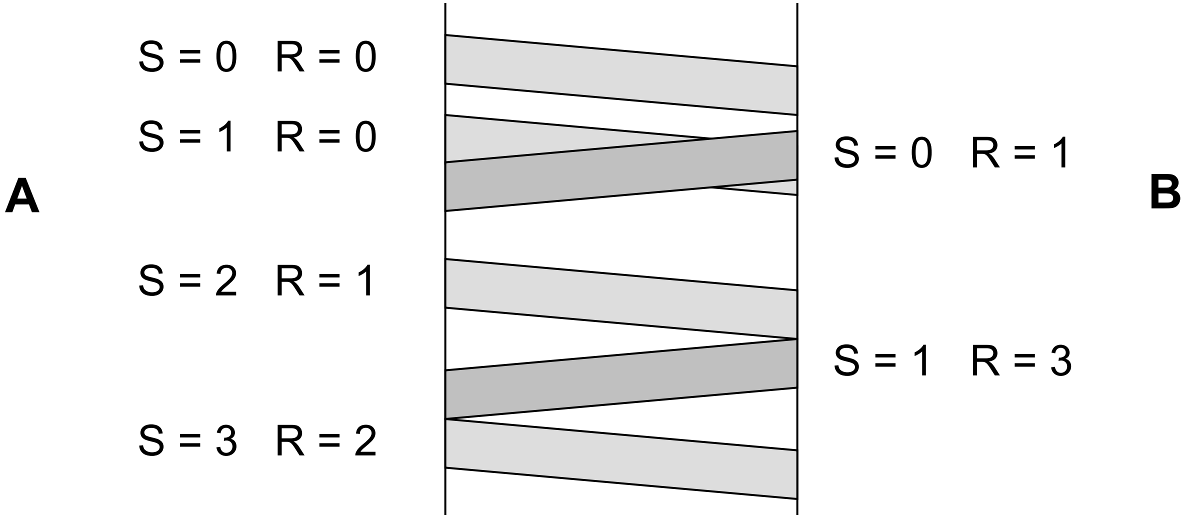
- I protocolli ARQ vengono utilizzati nello strato di linea ed in quello di trasporto in sinergia con una codifica a rivelazione di errore
- Obiettivo:
  - Rendere affidabile il canale di comunicazione
    - Affidabile?
      - Identifica errori di trasmissione e innesca la ritrasmissione
      - Riconosce perdita di informazioni
      - Riconosce perdite di sequenza
- Il canale tipicamente è:
  - Singolo collegamento seriale nello strato di linea
    - Flusso seriale di bit
  - Connessione end-to-end nello strato di trasporto
    - Cascata di nodi e collegamenti con diverse caratteristiche e prestazioni
- La diversità del canale rende le problematiche dei protocolli di trasporto più complesse ma esistono molti elementi in comune

# Controllo degli errori

- Alle PDU viene applicata una codifica di canale
- Il ricevitore
  - Verifica la correttezza delle PDU ricevute grazie a rivelazione di errore
  - Ignora le PDU errate
  - Può far partire le procedure di ritrasmissione
- Il trasmettitore
  - Ritrasmette le trame non correttamente ricevute
    - Su indicazione del ricevitore
    - Alla scadenza del time-out

# Numerazione

- I protocolli ARQ numerano sequenzialmente le unità informative (UI) da consegnare ai protocolli superiori
- Cosa numerare?
  - PDU
  - Unità informative standard (bit, byte …)
- Trasmettitore e ricevitore mantengono due contatori:
  - S conta in modo sequenziale le unità informative inviate
  - R conta le unità informative ricevute in modo corretto
- S permette il “posizionamento” nel flusso
- R permette la confermare di ricezione



# Conferma (Acknowledge)

- La corretta ricezione viene confermata dal ricevitore inviando al trasmettitore il proprio valore di R
  - Le PDU ricevute in modo corretto fanno aumentare R
  - Quando una PDU viene ricevuta in modo non corretto *viene ignorata* ed R *non viene modificato*
- La conferma della corretta ricezione può essere
  - Esplicita
    - Ogni PDU ricevuta correttamente genera una conferma
  - Implicita (cumulativa)
    - Una PDU di conferma con R = n conferma la ricezione fino a n-1
  - In *piggybacking*
    - Viaggia inserita (a "cavalluccio") in una PDU contenente dati utili

# Gli ACK

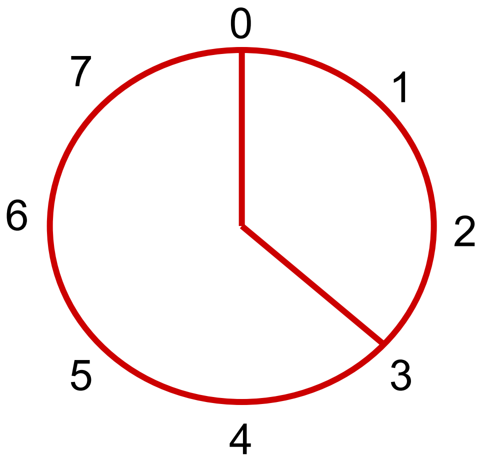
- Gli acknowledge o ACK
  - Sono PDU specializzate che non portano dati di utente ma solamente informazioni di controllo per il protocollo
- Servono qualora
  - Il protocollo ARQ non possa usare il piggybacking
  - Il ricevitore non abbia dati da trasmettere
- Non è necessario numerare gli ACK
  - I protocolli ARQ tipicamente
    - confermano la ricezione delle PDU che portano dati d’utente
    - non confermano la ricezione degli ACK (conferma della conferma)
  - Non si ritiene necessario controllare la sequenza degli ACK

# Finestra scorrevole

- Le funzioni di controllo
  - Dell’errore
  - Di flusso
  - Di sequenza
- Possono essere implementate con l’uso sinergico di
  - Codici di canale
  - Numerazione delle unità informative
  - Conferma di ricezione
- Il meccanismo utilizzato è quello della trasmissione a finestra scorrevole

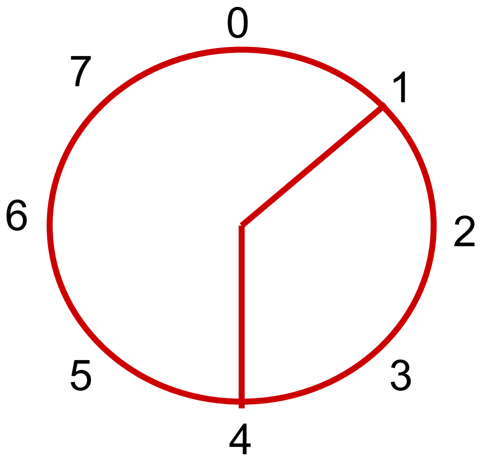
# Finestra di trasmissione

- $W_T$ = numero massimo di trame che il trasmettitore può inviare senza ricevere alcuna conferma
- La numerazione delle trame viene effettuata modulo $M$
  - $M = 2^n$ dove $n$ è il numero di bit utilizzati per la numerazione
- Si può procedere con la trasmissione di nuove trame solo al ricevimento delle conferme
  - La numerazione delle trame trasmesse scorre nel tempo (sliding window)



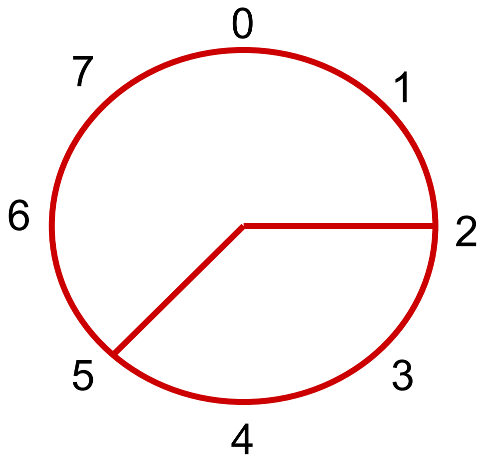
$W_T = 4$

Trasmetto 0, 1, 2, 3



Ricevuto ACK 1

Trasmetto 4



Ricevuto ACK 2

Trasmetto 5

# Dimensione della finestra

- Per quale motivo imporre W finito e sospendere la trasmissione delle trame?
  - Garantire unicità di numerazione delle trame
    - Lo spazio di numerazione
      - Dipende dal numero di bit dedicati alla numerazione nell'intestazione
      - Ha necessariamente dimensioni limitate
    - Se si continuasse a trasmettere all'infinito non si avrebbe più una corrispondenza biunivoca trame-numero
      - Le trame con uguale numerazione sono indistinguibili

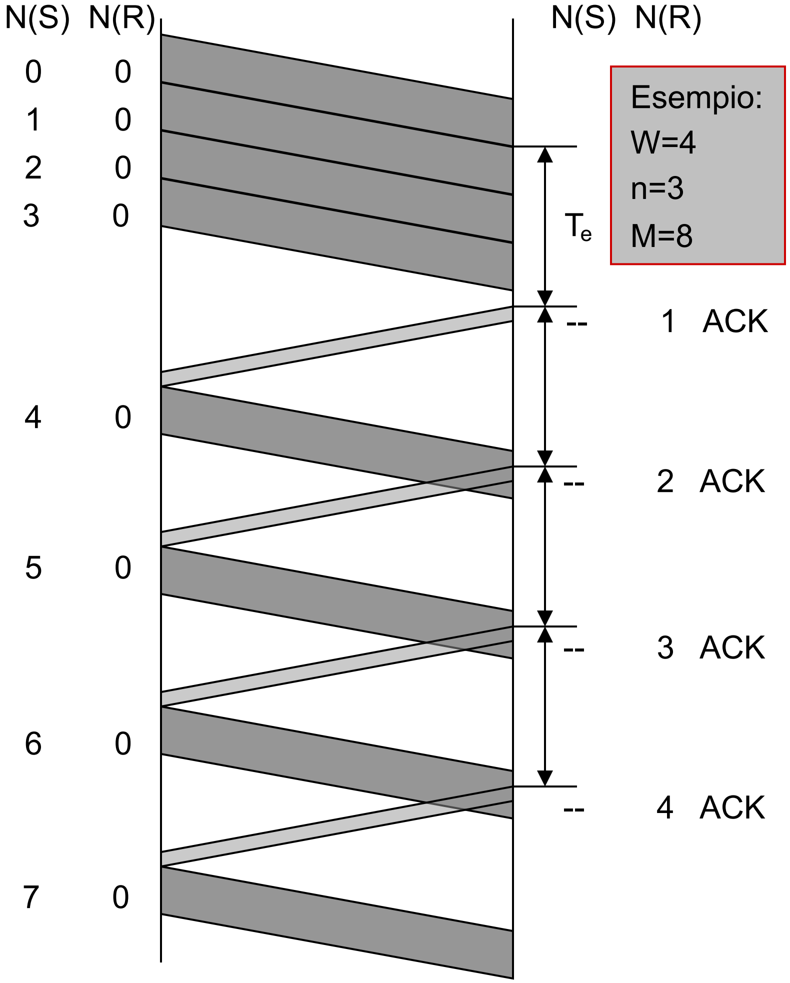
# Efficacia della numerazione a finestra

- Permettere la gestione automatica del controllo di flusso
  - Il ricevitore deve poter ricevere un’intera finestra, dopodichè “pilota” il trasmettitore con gli ACK
- Permette di riconoscere l’errata ricezione o la perdita di dati
  - Il ricevitore vede arrivare una trama (segmento) fuori sequenza
- Permette di ricostruire in ricezione la corretta sequenza dei dati

# Controllo di flusso

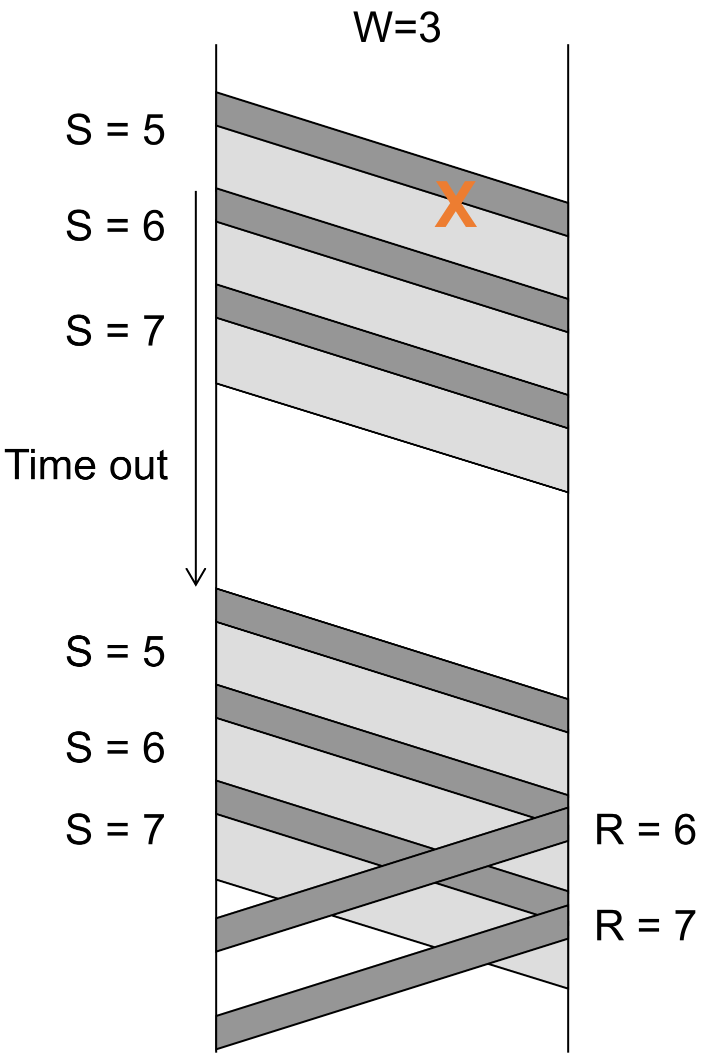
- Accorda la velocità del trasmettitore alla capacità del ricevitore (e della rete)

- Il ricevitore
  - Deve essere in grado di gestire un' intera finestra
    - Memorizzazione ed elaborazione di W trame
  - Accorda il flusso di trame in arrivo tramite le conferme
- A regime un nuova trama ogni $T_e$
  - $T_e$ = tempo necessario per elaborare una trama



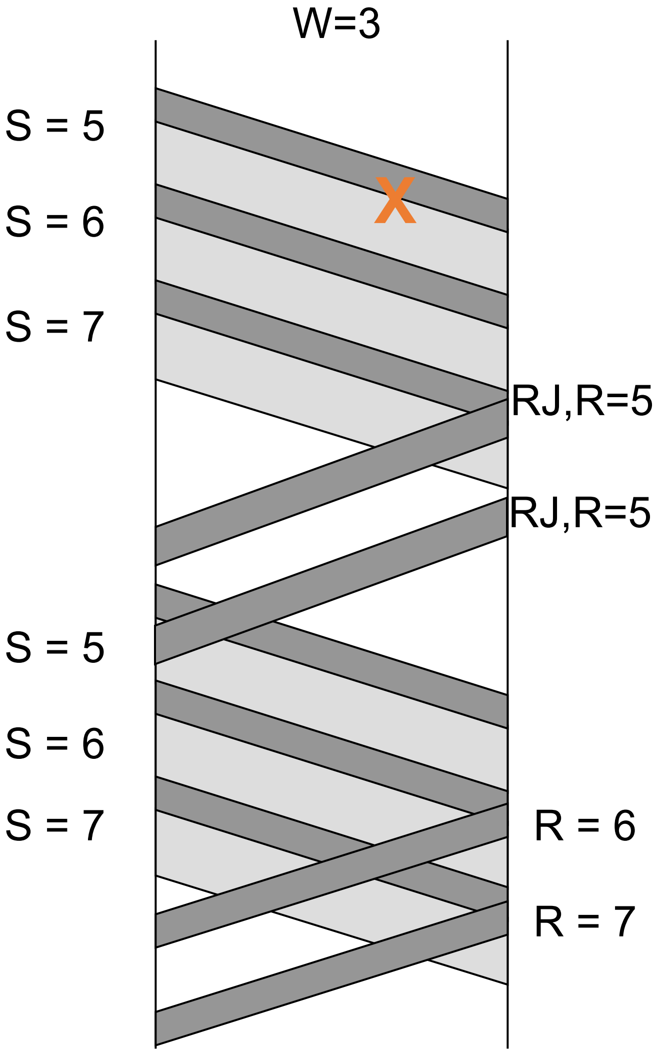
# Recupero dell’errore: go-back-n ARQ

- Viene persa la trama *N*
- Il ricevitore
  - Scarta tutte le trame successive a quella errata
  - A seconda dell’ implementazione
    - Segnale al trasmettitore la mancata ricezione della trama N
    - Rimane in silenzio senza inviare alcuna trama di segnalazione
- Il trasmettitore
  - Ritrasmette tutte la trame a partire dalla numero *N*
- Vantaggi
  - Semplicità operativa
  - Ridotta complessità nel ricevitore
- Svantaggi
  - Inefficienza
    - Si ritrasmettono trame senza che questo sia strettamente necessario



# Recupero dell’errore: go-back-n ARQ

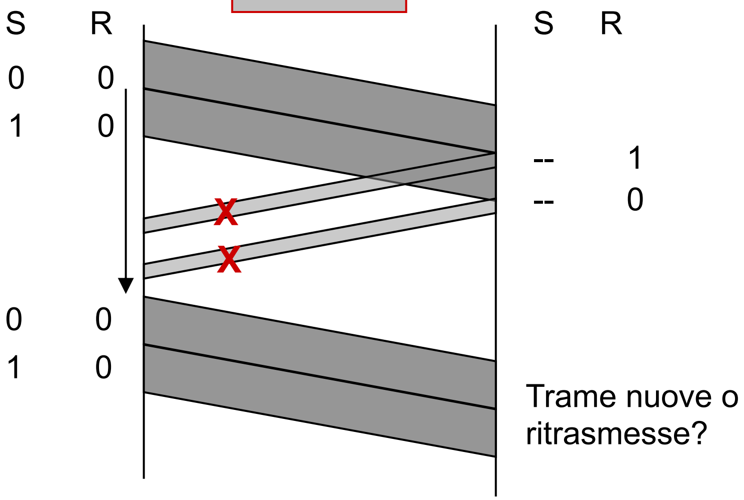
- Viene persa la trama *N*
- Il ricevitore
  - Scarta tutte le trame successive a quella errata
  - A seconda dell’ implementazione
    - Segnale al trasmettitore la mancata ricezione della trama N
    - Rimane in silenzio senza inviare alcuna trama di segnalazione
- Il trasmettitore
  - Ritrasmette tutte la trame a partire dalla numero *N*
- Vantaggi
  - Semplicità operativa
  - Ridotta complessità nel ricevitore
- Svantaggi
  - Inefficienza
    - Si ritrasmettono trame senza che questo sia strettamente necessario



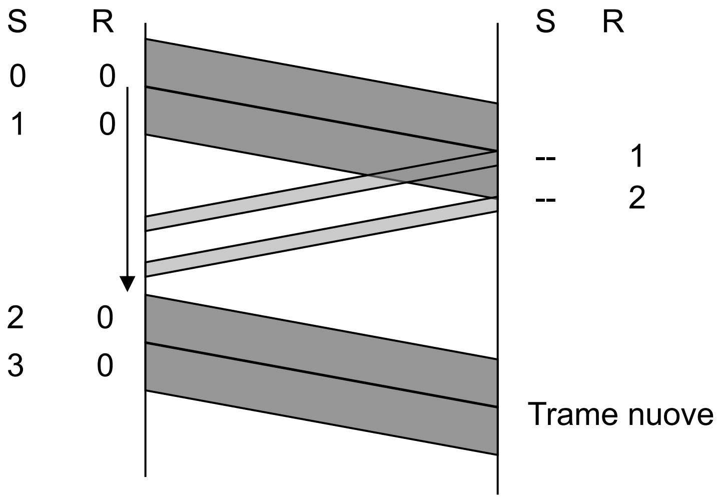
# Finestra e numerazione

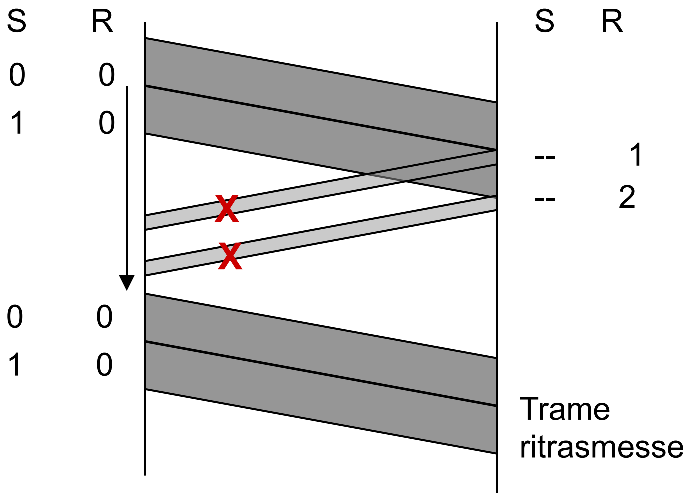
- Campo di numerazione finito ($n$ bit -> $M = 2^n$ diversi numeri di sequenza)
  - deve essere $W_T \leq M-1$

Esempio:
W=2
n=1
M=2



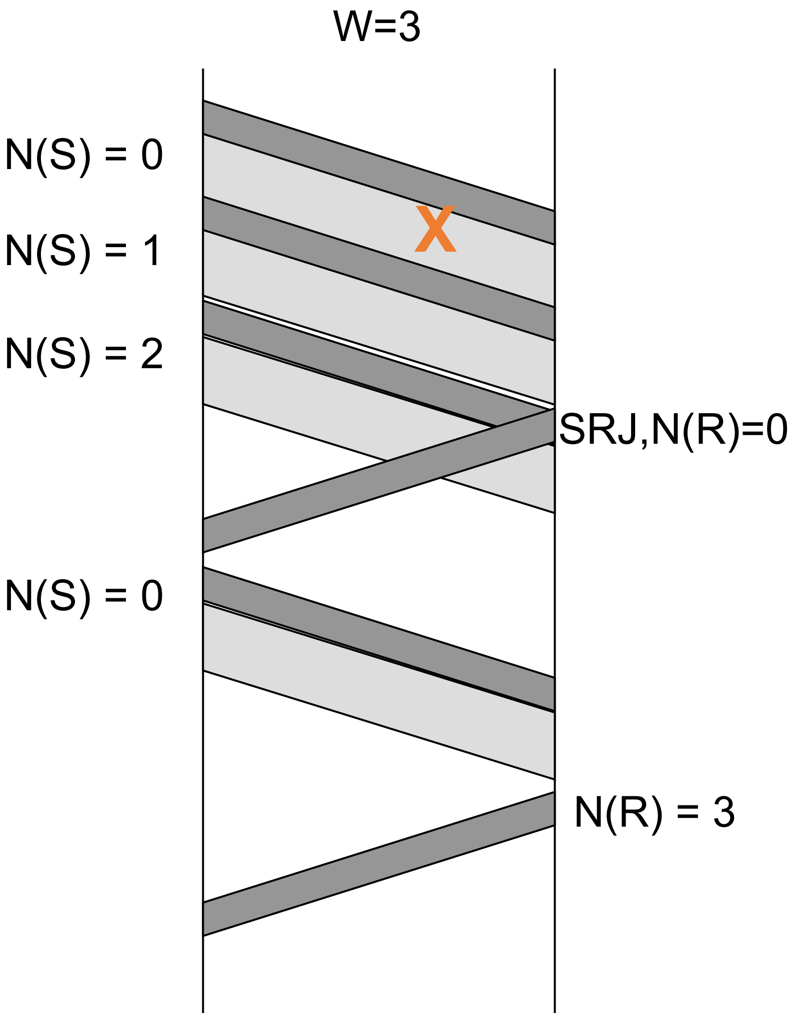
Esempio:
W=2
n=2
M=4





# Selective repeat ARQ

- Viene persa la trame *N*
- Il ricevitore
  - Scarta solamente la trama errata
  - Segnala la mancata ricezione della trama N
- Il trasmettitore
  - Ritrasmette solamente la trama *N*
- Il ricevitore
  - Riordina le trame nella memoria di ricezione
- Vantaggi
  - Maggiore efficienza
- Svantaggi
  - Complessità del ricevitore
    - Deve tenere in memoria le trame correttamente ricevute fintanto che non può consegnarle allo strato superiore nella giusta sequenza



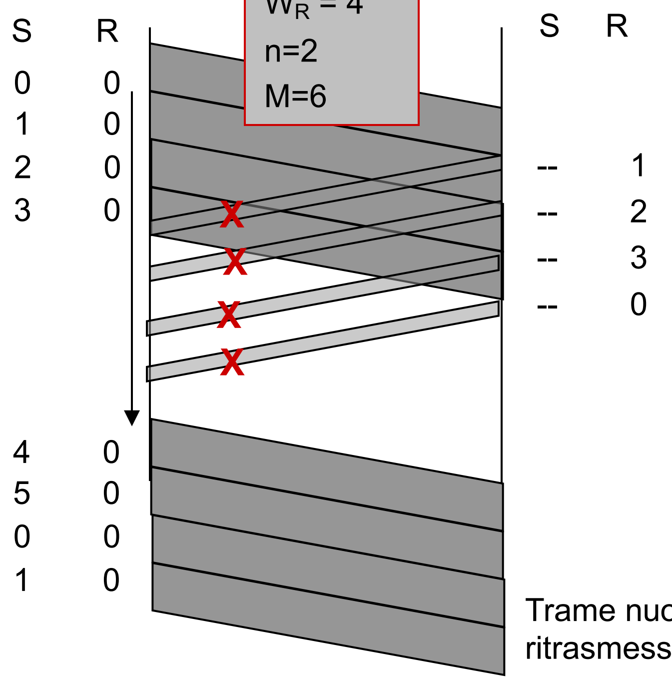
# Finestra di ricezione

- Il concetto di finestra si può applica anche in ricezione
- Finestra di ricezione
  - $W_R$ = massimo numero di PDU che Rx può memorizzare prima di consegnare i dati allo strato superiore
    - Nel protocollo go-back-N tipicamente $W_R = 1$
    - In un protocollo Selective Repeat deve essere $W_R > 1$

# Finestra e numerazione

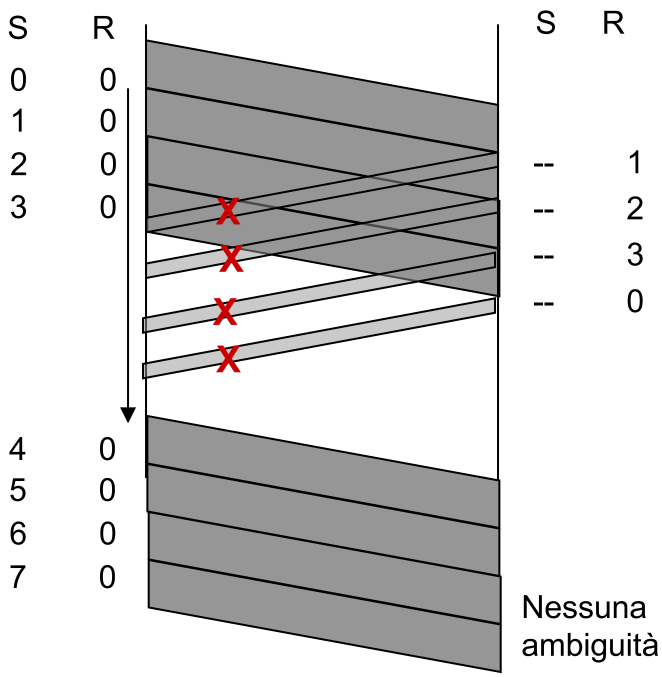
- Campo di numerazione finito ($n$ bit -> $M = 2^n$ diversi numeri di sequenza)
  - Se $W_T = W_R$ deve deve essere $W_T + W_R \leq M$

Esempio:
$W_T = 4$
$W_R = 4$
n=2
M=6



Trame nuove o ritrasmesse?

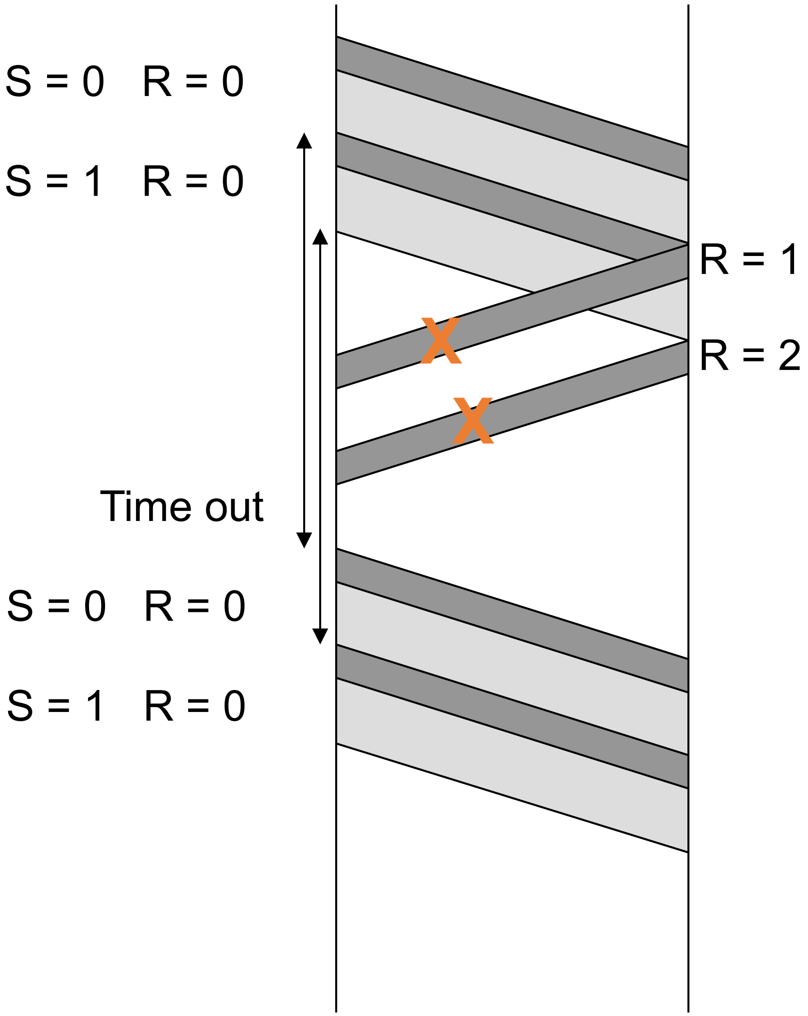
Esempio:
$W_T = 4$
$W_R = 4$
n=3
M=8



Nessuna ambiguità

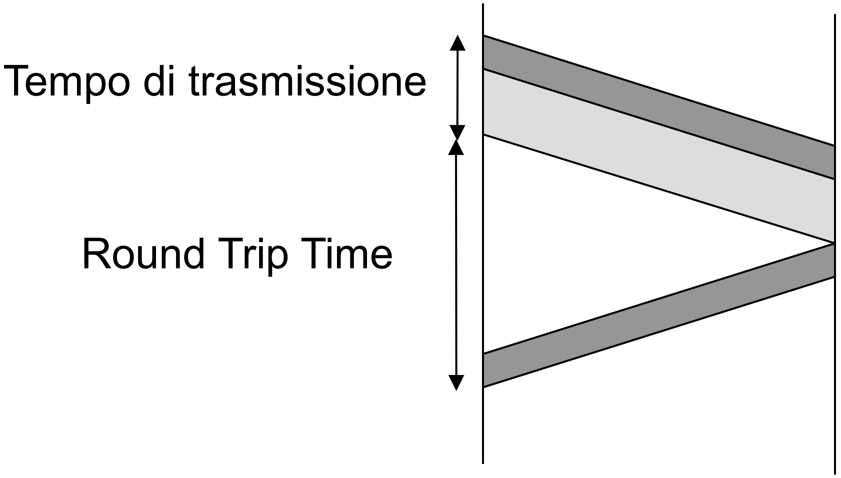
# Time out

- Il protocollo può entrare in stallo (deadlock)
  - Se le trame informative sono perdute
  - Se gli ACK sono perduti
- È necessario un time out per riprendere il dialogo
  - Un orologio parte al termine della trasmissione di ciascuna trama
  - Se si raggiunge il time out senza avere conferma si ritrasmette la trama



# Il round trip time (RTT)

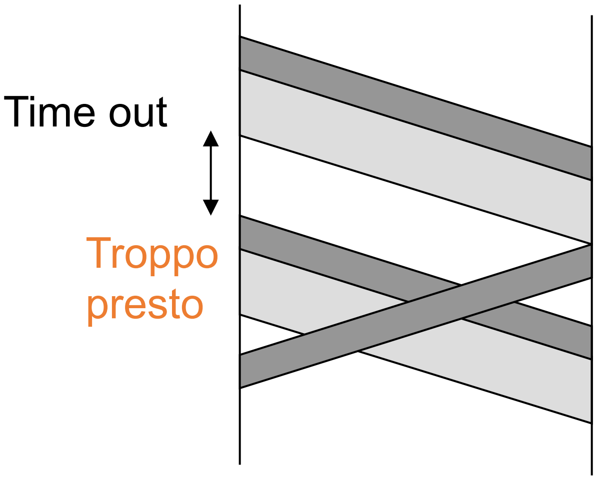
- RTT = tempo necessario per effettuare un' andata e ritorno sul canale
  - Tempo intercorso fra la partenza dell' ultimo bit di una trama e la ricezione del relativo ACK
- Variabilità di RTT
  - RTT è praticamente deterministico per lo strato 2
  - RTT può variare da segmento a segmento per lo strato 4



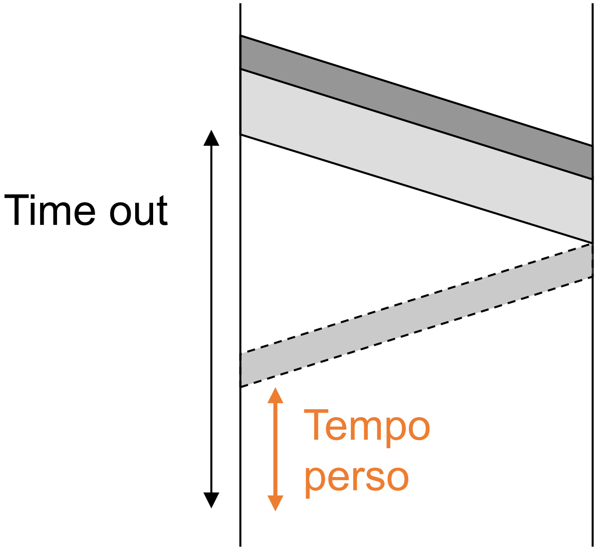
# Dimensione del Time out?

- Il time out va relazionato al RTT

- Time out troppo breve
  - Non si attende l’arrivo dell’ACK
  - Invio non necessario di trame duplicate

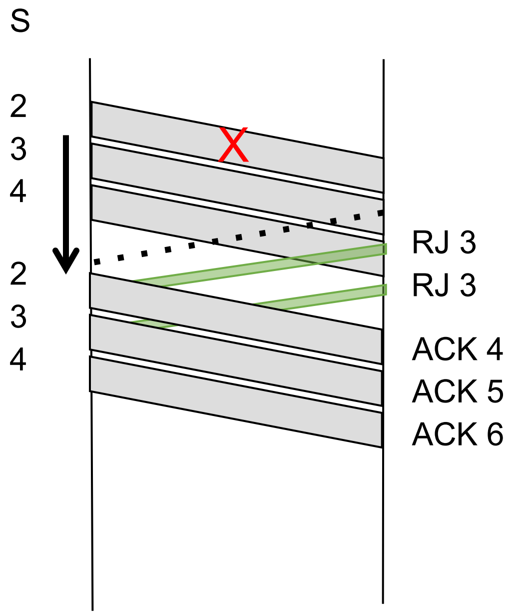


- Time out troppo lungo
  - Inutile attesa prima di ritrasmettere le trame errate

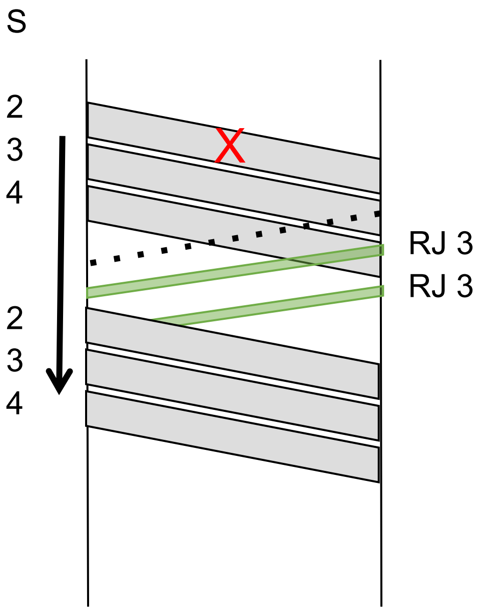


- In entrambi i casi
  - Si spreca capacità di trasmissione (banda)
  - Degradano le prestazioni

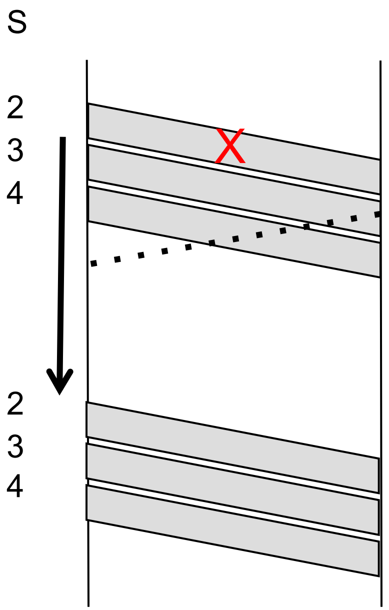
# Esempio: $W_T = 3$



Time out correttamente dimensionato: equivalente con o senza Reject



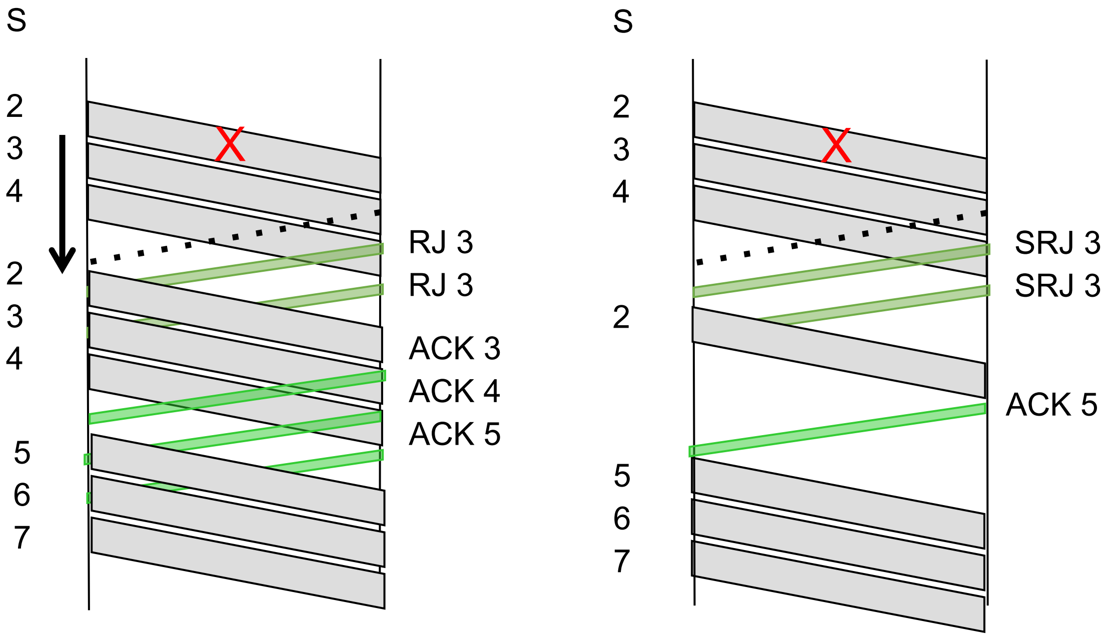
Time out mal dimensionato: Reject permette di reagire prima alla perdita



Time out mal dimensionato: l’assenza di Reject fa perdere tempo

- Go-Back-N con e senza ACK negativo (Reject)
  - Reject utile in case di timeout mal dimensionato

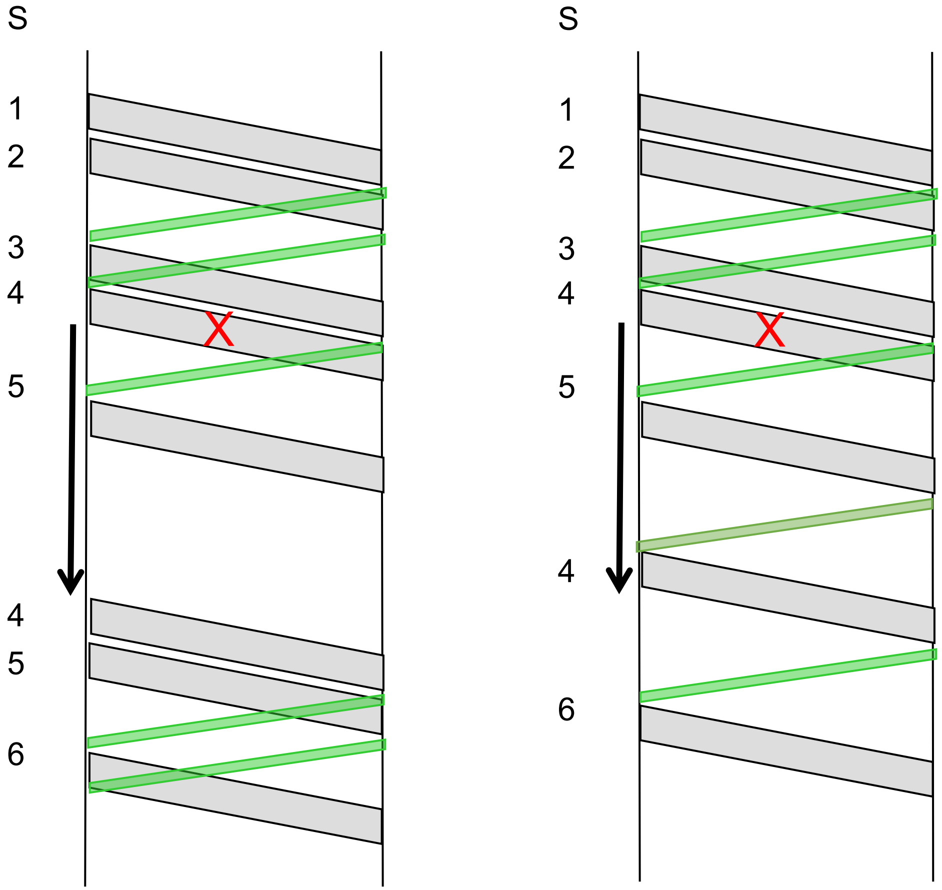
# Esempio: $W_T = 3$



La dimensione della finestra è prossima al RTT

- Go-Back-N a confronto con Selective Repeat
  - Una perdita singola non determina particolare differenza

# Esempio: $W_T = 2$

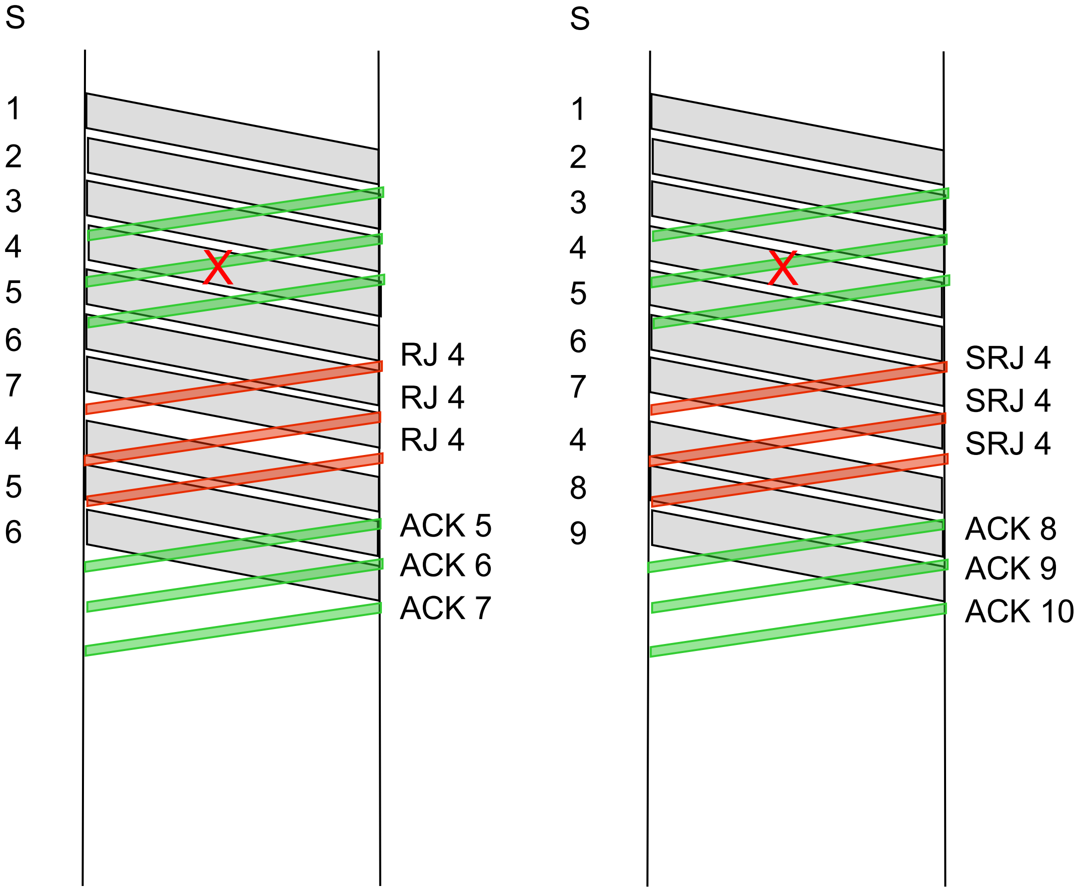


Il time out è sovradimensionato

$W_T < RTT$

- Selective Repeat ha un vantaggio

# Esempio: $W_T = 6$



- Selective Repeat ha un vantaggio
  - Una perdita singola non determina particolare differenza