

Livello di trasporto: meccanismi trasferimento dati affidabile

Gaia Maselli

Queste slide sono un adattamento delle slide fornite dal libro di testo e pertanto protette da copyright.

- Copyright © 2013 McGraw-Hill Education Italy srl

- All material copyright 1996-2007 J.F Kurose and K.W. Ross, All Rights Reserved

Stop-and-wait

- ❑ Orientato alla connessione + Controllo di flusso + Controllo degli errori
- ❑ Mittente e destinatario usano una finestra scorrevole di dimensione 1
- ❑ Il mittente invia un pacchetto alla volta e ne attende l'ack prima di spedire il successivo
- ❑ Quando il pacchetto arriva al destinatario viene calcolato il checksum
 - Pacchetto corretto → ack al mittente
 - Pacchetto corrotto → scartato senza informare il mittente
- ❑ Per capire se un pacchetto è andato perso il mittente (non riceve ack e non può aspettare all'infinito) usa un timer
 - Scadenza timer senza ricevere ack → rinvio pacchetto

Stop and wait



impossibile visualizzare l'immagine. La memoria del computer potrebbe essere insufficiente per aprire l'immagine oppure l'immagine potrebbe essere danneggiata. Riavviare il computer e aprire di nuovo il file. Se viene visualizzata di nuovo la x rossa, potrebbe essere necessario eliminare l'immagine e inserirla di nuovo.



impossibile visualizzare l'immagine. La memoria del computer potrebbe essere insufficiente per aprire l'immagine oppure l'immagine potrebbe essere danneggiata. Riavviare il computer e aprire di nuovo il file. Se viene visualizzata di nuovo la x rossa, potrebbe essere necessario eliminare l'immagine e inserirla di nuovo.

- ❑ Il mittente deve tenere una copia del pacchetto spedito finché non riceve riscontro
- ❑ Controllo errori (mediante numero sequenza + ack + timer)
- ❑ Controllo di flusso (non si spedisce più di un pacchetto alla volta)

Numeri di sequenza e riscontro nello stop and wait

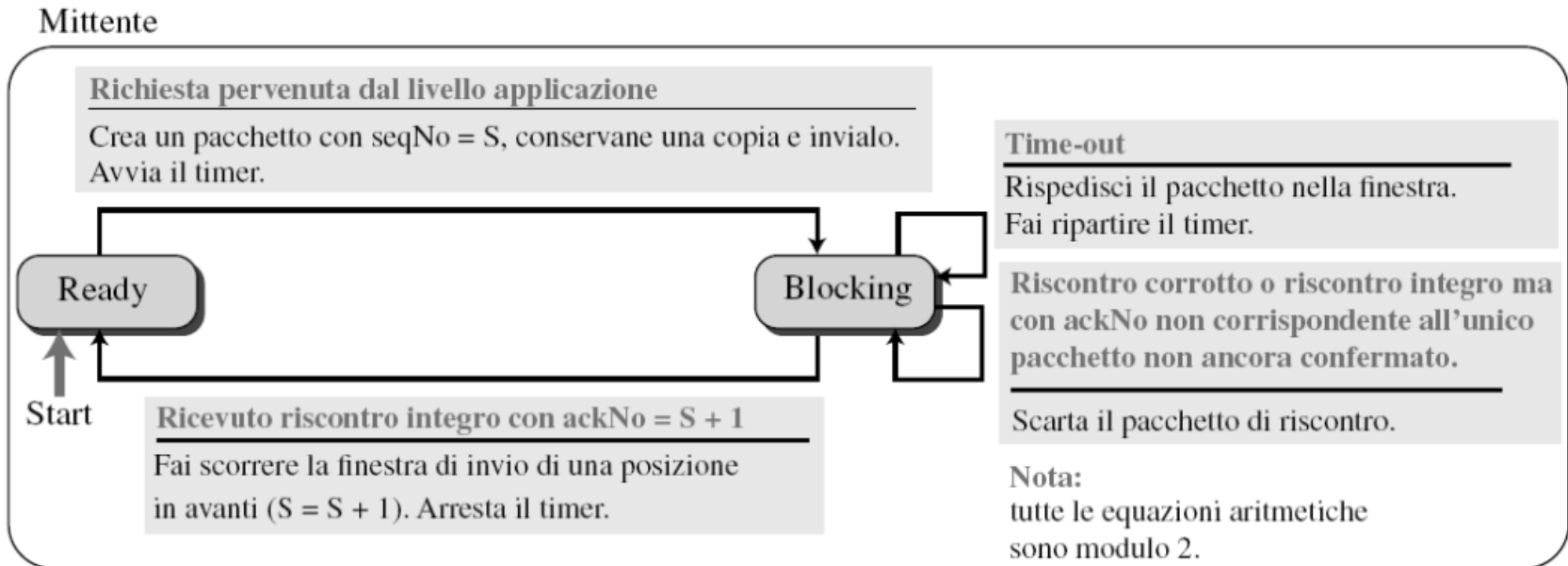
- Qual è l'intervallo minimo dei numeri di sequenza per lo stop and wait?
- $x, x+1, x+2, \dots$?
- Supponiamo che il mittente abbia inviato il pacchetto con numero di sequenza x . Si possono verificare 3 casi:
 1. Il pacchetto arriva correttamente al destinatario che invia un riscontro. Il riscontro arriva al mittente che invia il pacchetto successivo numerato $x+1$
 2. Il pacchetto risulta corrotto o non arriva al destinatario. Il mittente allo scadere del timer invia nuovamente il pacchetto x
 3. Il pacchetto arriva correttamente al destinatario ma il riscontro viene perso o corrotto. Scade il timer e il mittente rispedisce x . Il destinatario riceve un duplicato, se ne accorge?

Numeri di sequenza e riscontro nello stop and wait

- ❑ I numeri di sequenza 0 e 1 sono sufficienti per il protocollo stop and wait
- ❑ Convenzione: il numero di riscontro (ack) indica sempre il numero di sequenza del prossimo pacchetto atteso dal destinatario
 - Se il destinatario ha ricevuto correttamente il pacchetto 0 invia un riscontro con valore 1 (che significa che il prossimo pacchetto atteso ha numero di sequenza 1)

Nel meccanismo stop and wait il numero di riscontro indica, in aritmetica modulo 2, il numero di sequenza del prossimo pacchetto atteso dal destinatario

FSM mittente per Stop-and-wait



Una volta inviato un pacchetto il mittente si blocca (non spedisce il pacchetto successivo) e aspetta (stop and wait) finche' non riceve l'ack

FSM destinatario per Stop-and-wait

Il destinatario è sempre nello stato ready

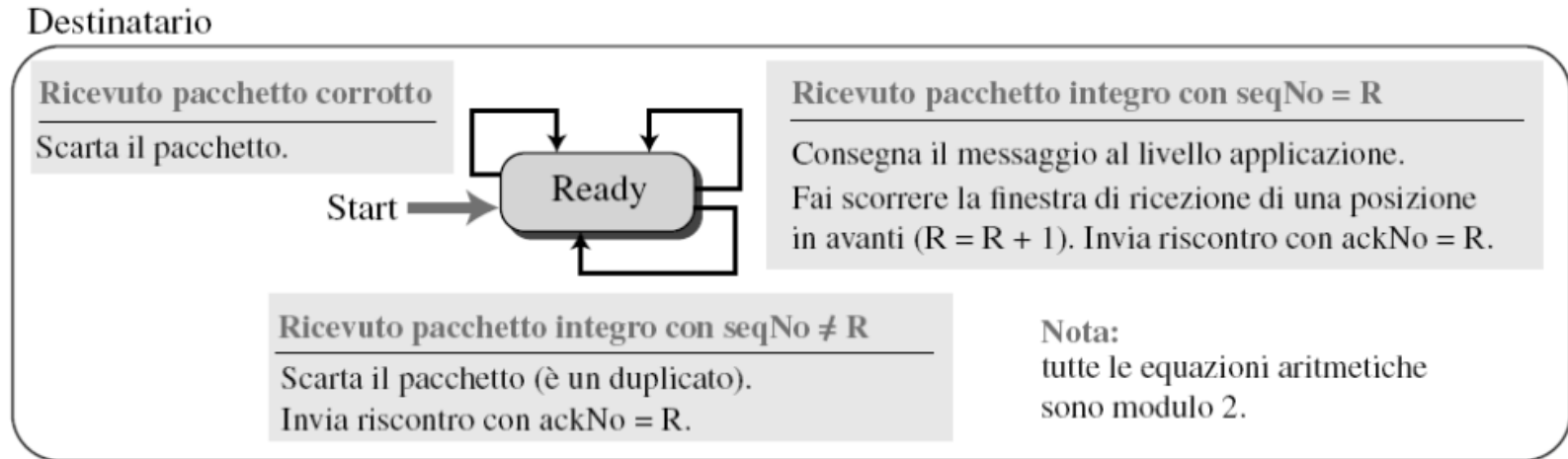
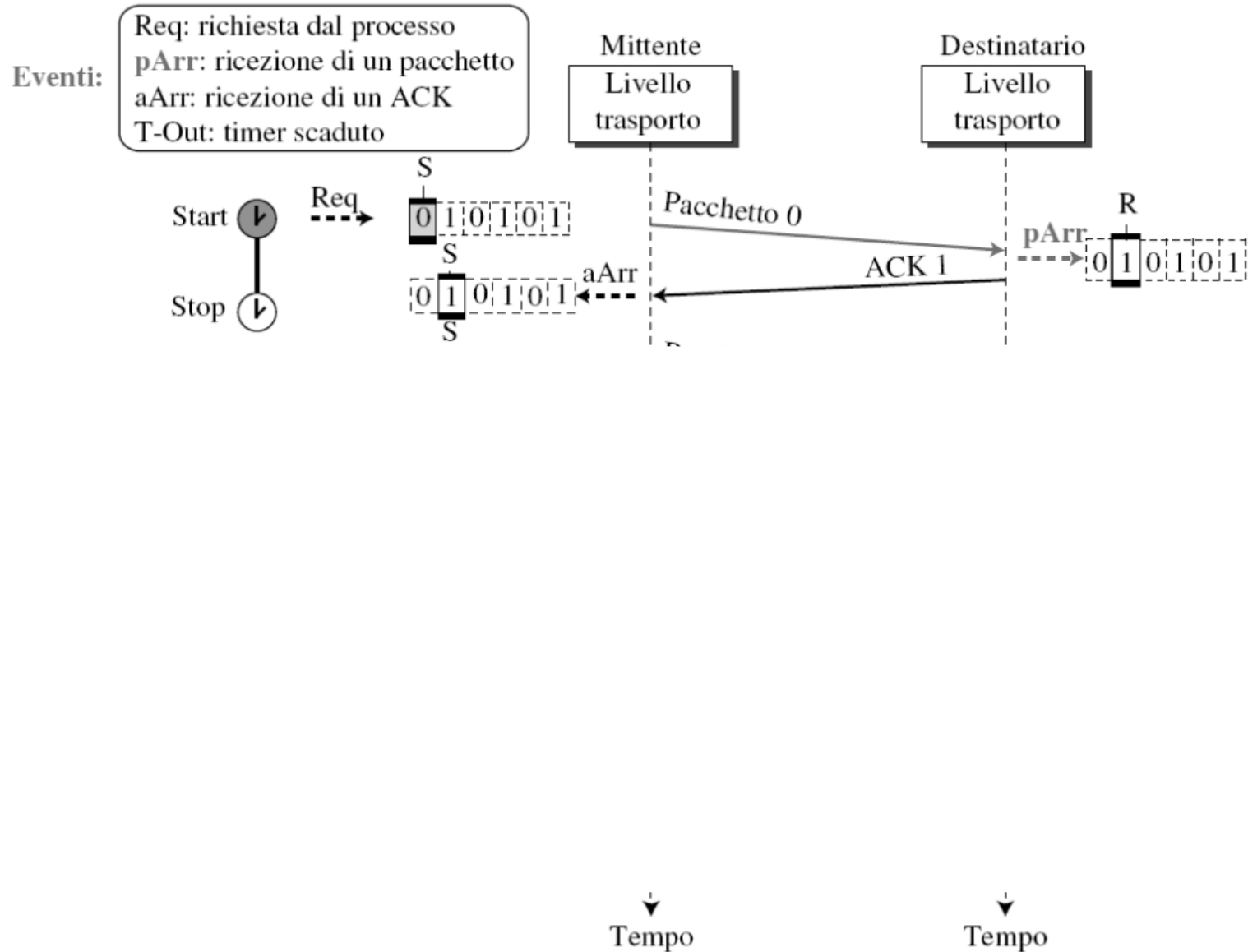


Diagramma di comunicazione



Efficienza di stop-and-wait

- Consideriamo il prodotto $\text{rate} \times \text{ritardo}$ (misura del numero di bit che il mittente può inviare prima di ricevere un ack, volume della pipe in bit)
- Se rate elevato e ritardo lungo
 - stop and wait inefficiente!

esempio

In un sistema che utilizza Stop and Wait abbiamo

- Rate = 1Mbps

- Ritardo di andata e ritorno di 1 bit = 20 ms

Quanto vale rate*ritardo?

Se i pacchetti hanno dimensione 1000bit, qual è la percentuale di utilizzo del canale?

Rate*ritardo = $(1 \times 10^6) \times (20 \times 10^{-3}) = 20000$ bit

Il mittente potrebbe inviare 20000 bit nel tempo necessario per andare dal mittente al ricevente e viceversa ma ne invia solo 1000.

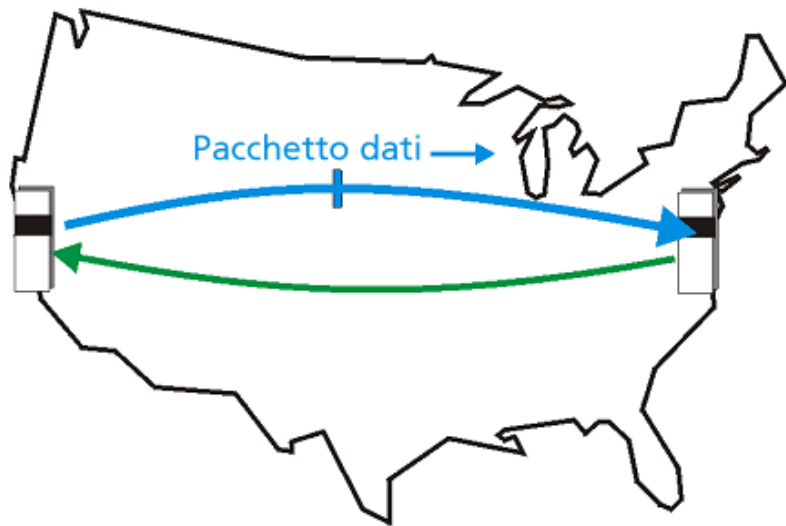
Il coefficiente di utilizzo del canale è $1000/20000 = 5\%$

→ molto inefficiente

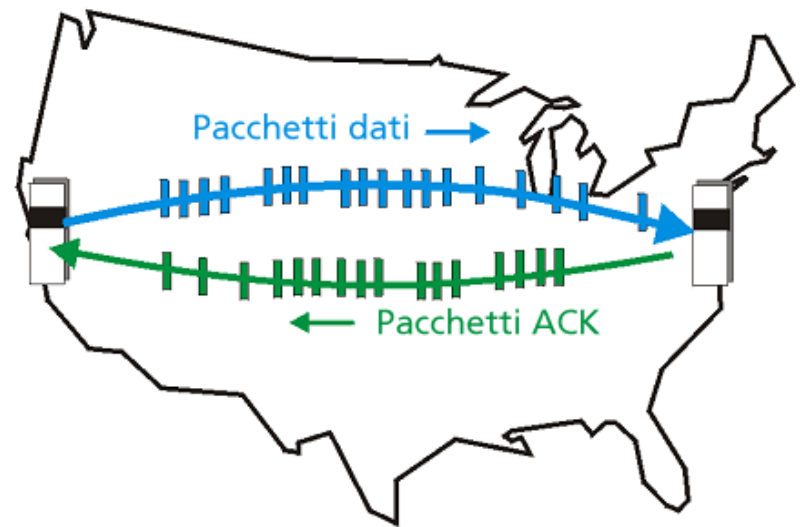
Protocolli con pipeline

Pipelining: il mittente ammette più pacchetti in transito, ancora da notificare

- l'intervallo dei numeri di sequenza deve essere incrementato
- buffering dei pacchetti presso il mittente e/o ricevente



a) Protocollo stop-and-wait all'opera

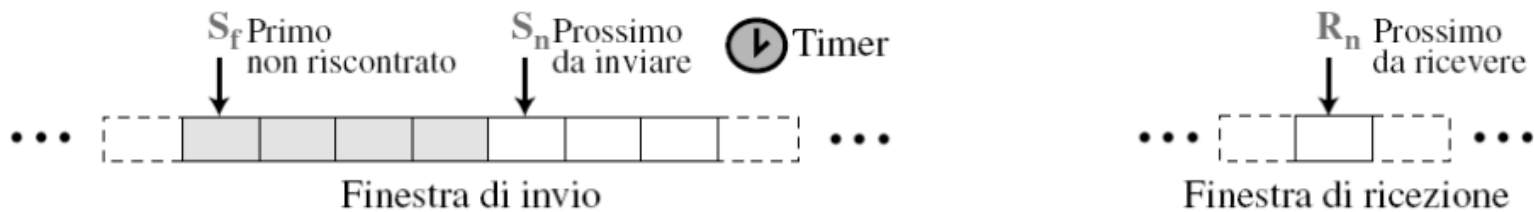
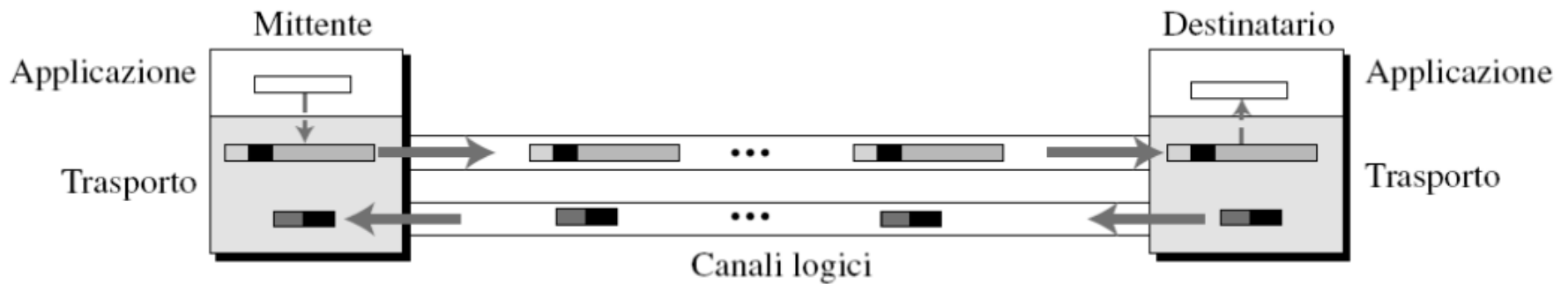
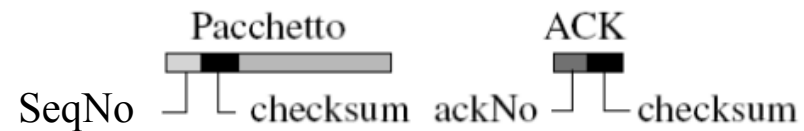


b) Protocollo con pipeline all'opera

- Due forme generiche di protocolli con pipeline:
Go-Back-N e *ripetizione selettiva*

Go back N

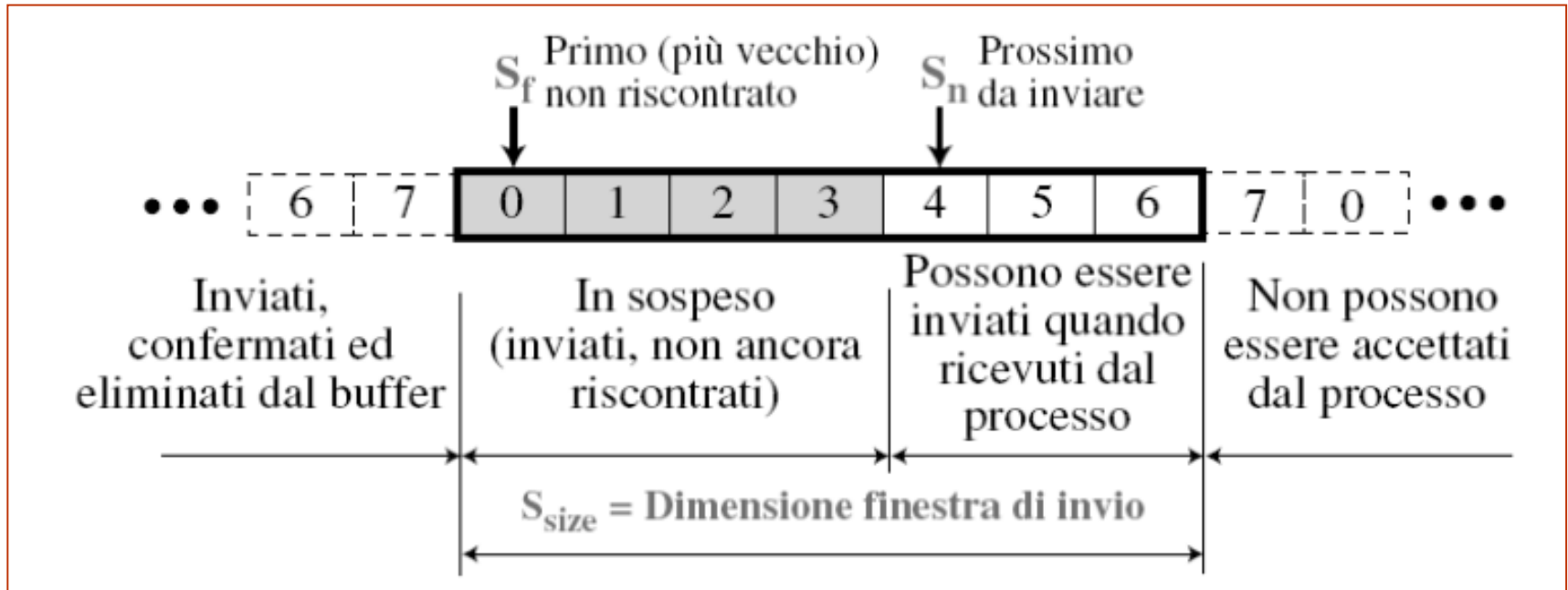
Schema generale



Numeri di sequenza e riscontro

- ❑ I numeri di sequenza sono calcolati modulo 2^m , dove m è la dimensione del campo "numero di sequenza" in bit
- ❑ Ack indica il numero di sequenza del prossimo pacchetto atteso
- ❑ **Ack cumulativo**: tutti i pacchetti fino al numero di sequenza indicato nell'ack sono stati ricevuti correttamente
 - Esempio AckNo=7, i pacchetti fino al 6 sono stati ricevuti correttamente e il destinatario attende il 7

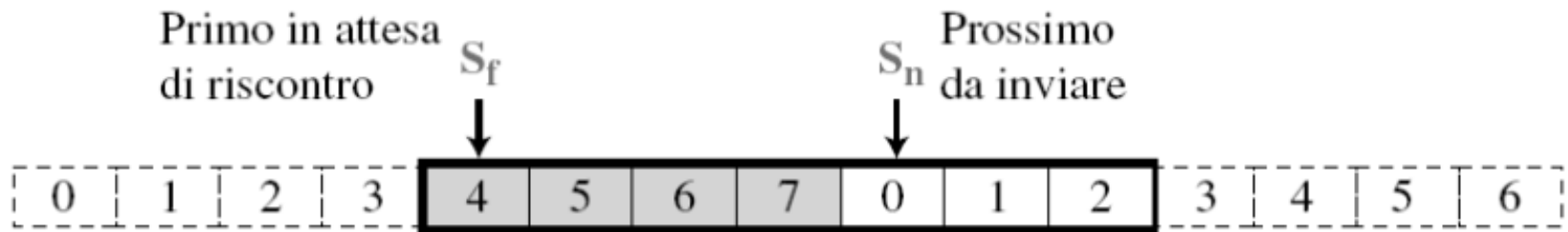
Finestra di invio



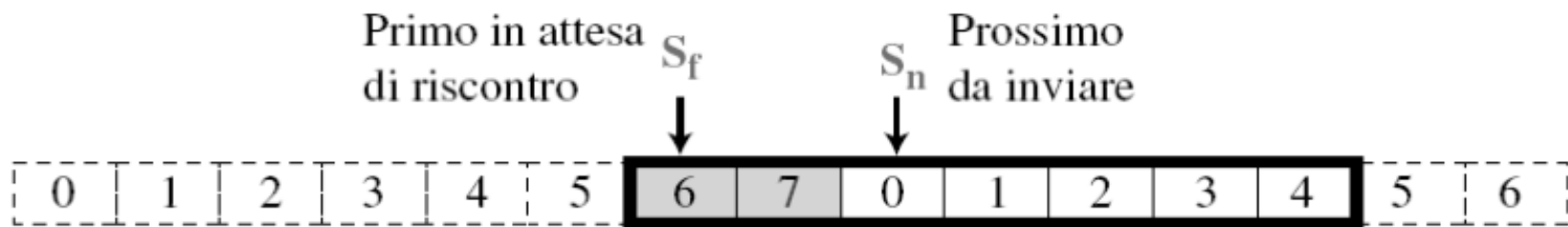
La finestra di invio è un concetto astratto che definisce una porzione immaginaria di dimensione massima $2^m - 1$ con tre variabili, S_f , S_n , S_{size} .

La finestra di invio può scorrere uno o più posizioni quando viene ricevuto un riscontro privo di errori con $ackNo$ maggiore o uguale a S_f e, minore di S_n in aritmetica modulare

Esempio di scorrimento della finestra di invio



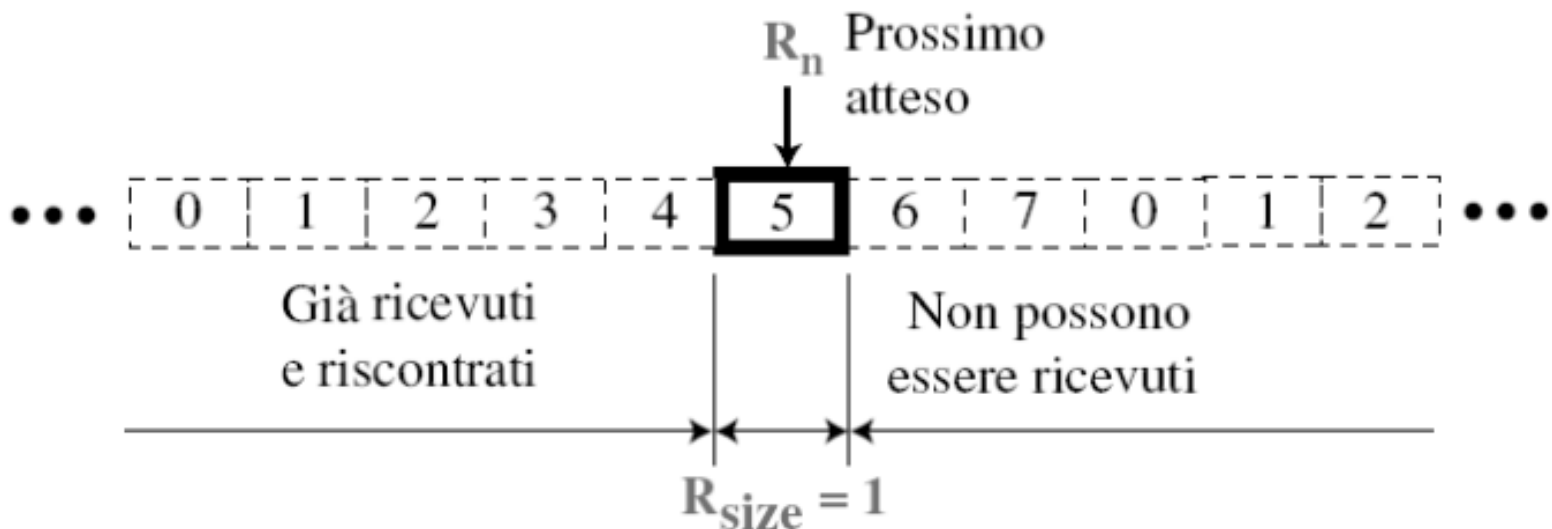
a. Finestra prima dello scorrimento.



b. Finestra dopo lo scorrimento (è arrivato un ACK con ackNo = 6).

Finestra di ricezione

- ❑ La finestra di ricezione ha dimensione 1
- ❑ Il destinatario è sempre in attesa di uno specifico pacchetto, qualsiasi pacchetto arrivato fuori sequenza (appartenente alle due regioni esterne alla finestra) viene scartato



La finestra di ricezione può scorrere di una sola posizione: $R_n = (R_n + 1) \bmod 2^m$

Timer e rispedizione

- ❑ Il mittente mantiene un timer per il più vecchio pacchetto non riscontrato
- ❑ Allo scadere del timer, Go back N, ovvero vengono rispediti tutti i pacchetti in attesa di riscontro (il destinatario ha finestra di ricezione pari a 1 e non può bufferizzare i pacchetti fuori sequenza)
- ❑ Esempio: $S_f=3$ e il mittente ha inviato il pacchetto 6 ($S_n = 7$). Scade il timer, allora i pacchetti 3,4,5,6 non sono stati riscontrati e devono essere rispediti

FSM mittente

Mittente

Nota:

tutte le equazioni aritmetiche sono modulo 2^m .

Time-out

Rispedisci tutti i pacchetti in attesa di riscontro.
Riavvia il timer.

Ricevuto messaggio dal processo

Costruisci un pacchetto ($seqNo = S_n$).
Memorizzane una copia e invia il pacchetto.
Avvia il timer se non è già attivo.
 $S_n = S_n + 1$.

Time-out

Rispedisci tutti i pacchetti in attesa di riscontro.
Riavvia il timer.

Finestra piena
($S_n = S_f + S_{size}$)?



FSM destinatario

Destinatario

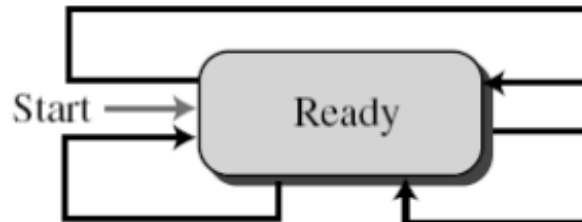
Nota:
tutte le equazioni
aritmetiche
sono modulo 2^m .

Ricevuto pacchetto integro con
 $\text{seqNo} = R_n$

Consegna il messaggio.
Fai scorrere la finestra ($R_n = R_n + 1$).
Invia un ACK ($\text{ackNo} = R_n$).

Ricevuto pacchetto corrotto

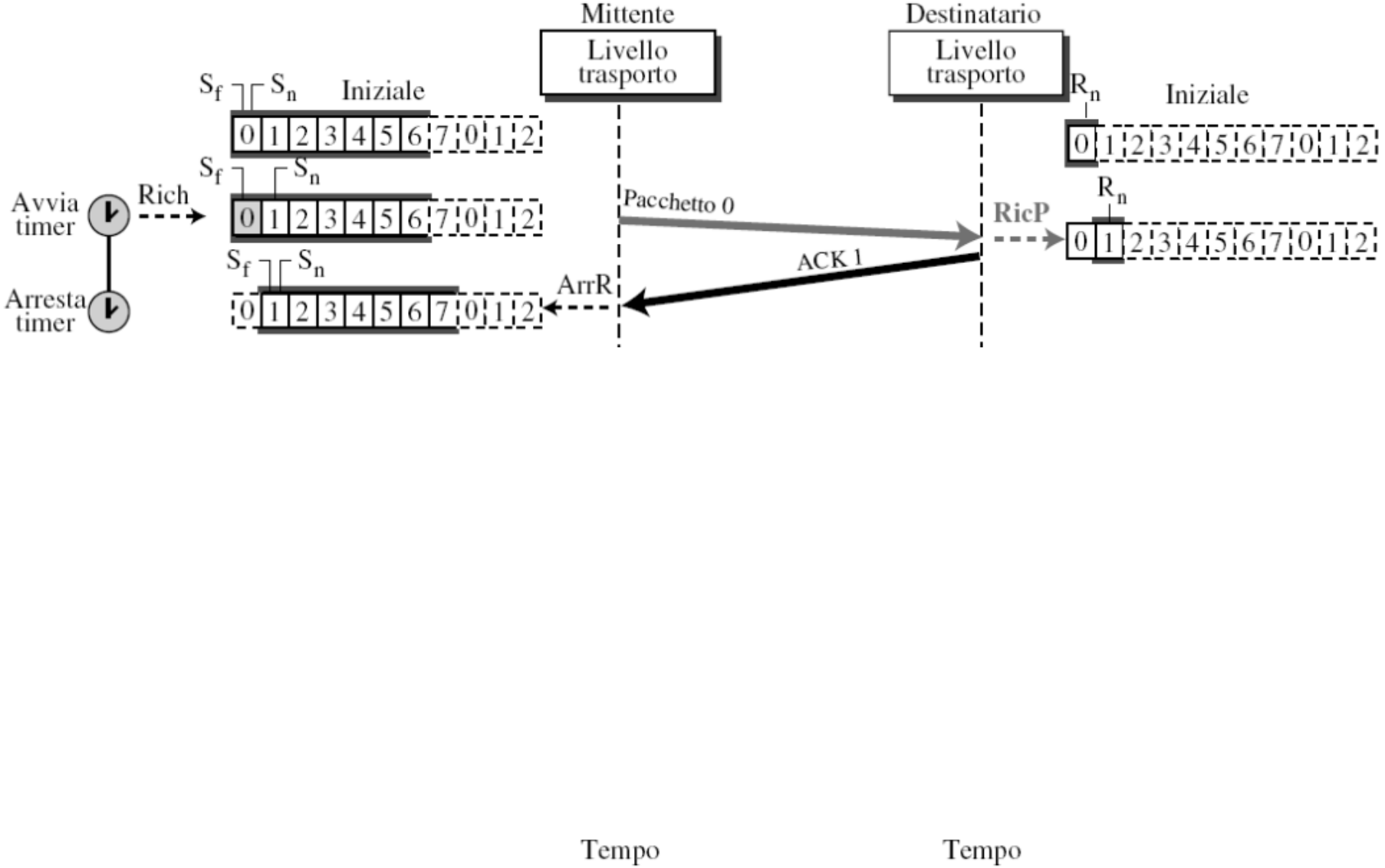
Scarta il pacchetto.



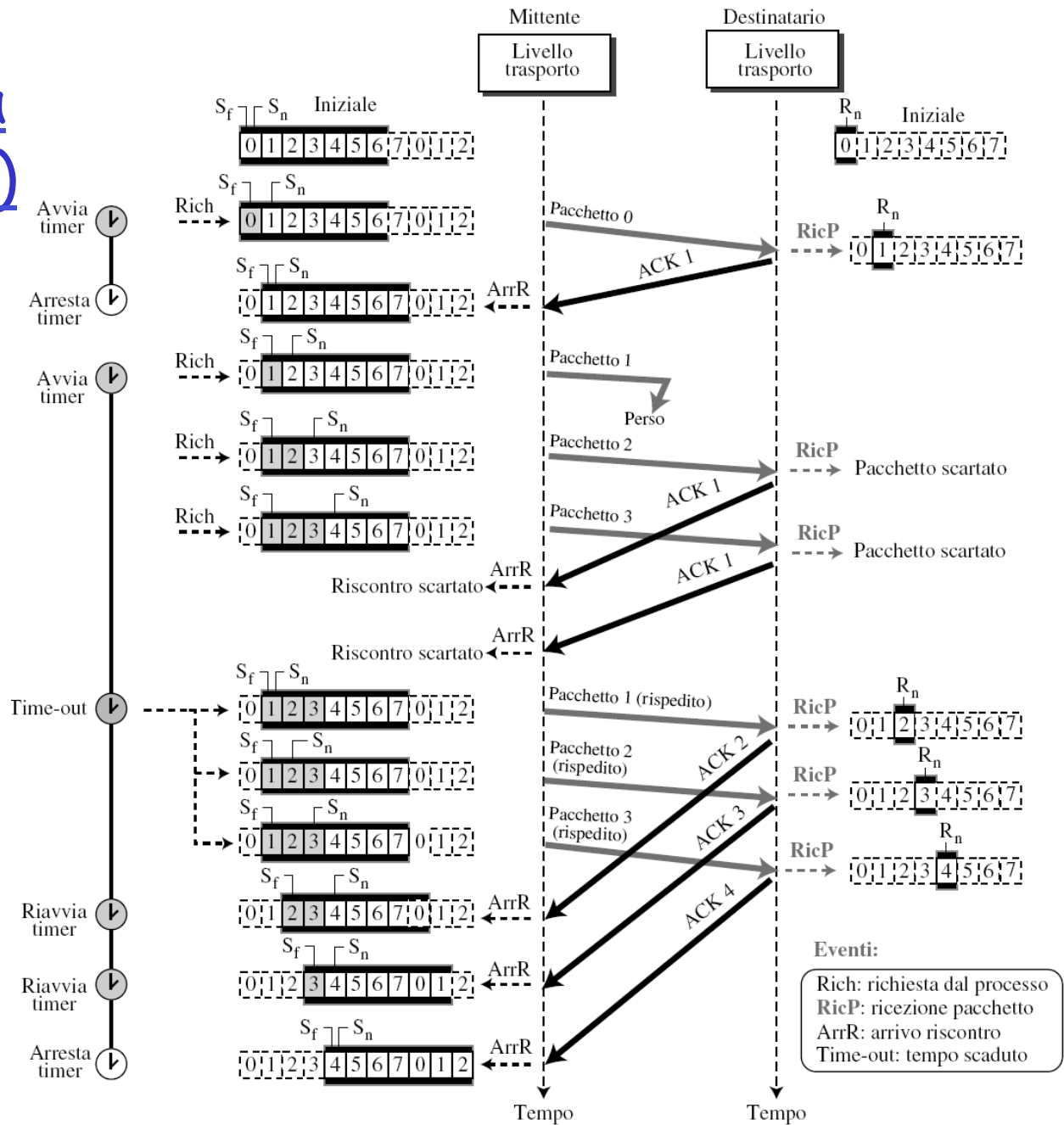
Ricevuto pacchetto integro
con $\text{seqNo} \neq R_n$

Scarta il pacchetto.
Invia un ACK ($\text{ackNo} = R_n$).

Go back N in azione (ack cumulativo)



Go back N in azione (perdita pacchetto dati)

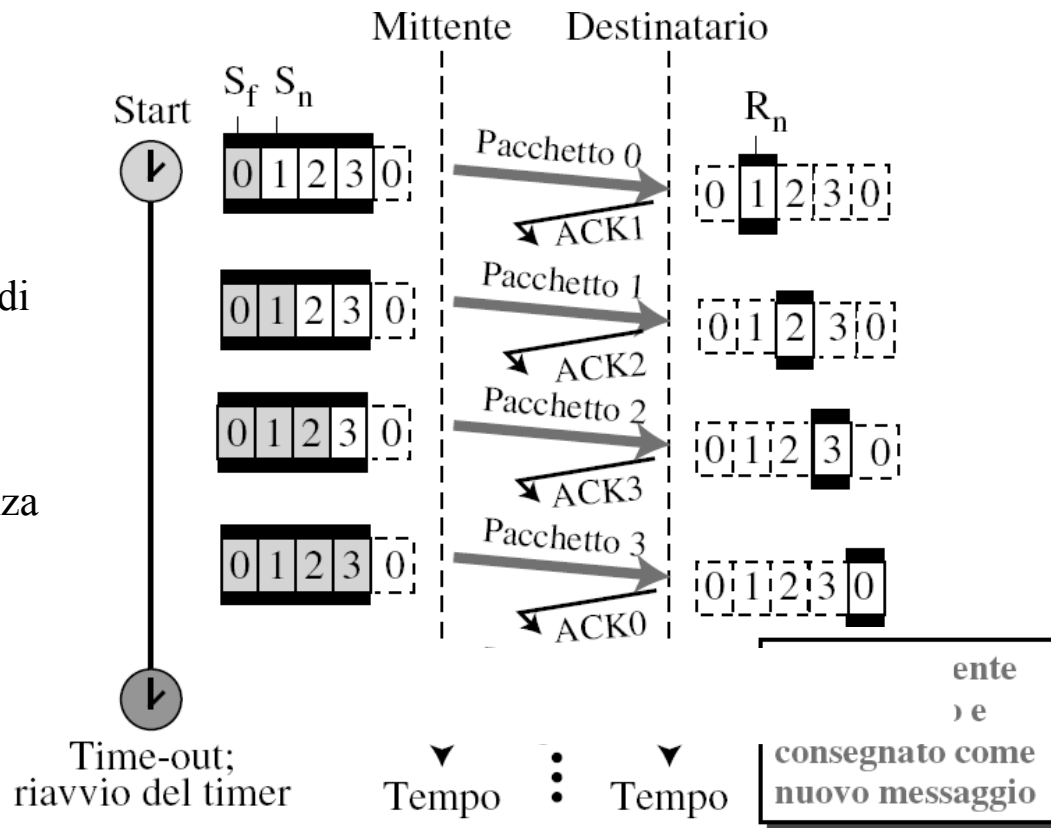


Dimensione finestra di invio

- Qual è la relazione fra lo spazio dei numeri di sequenza e la dimensione della finestra di invio?
- Possiamo avere una finestra di dimensione 2^m ?

m: numero di bit per rappresentare numero di sequenza

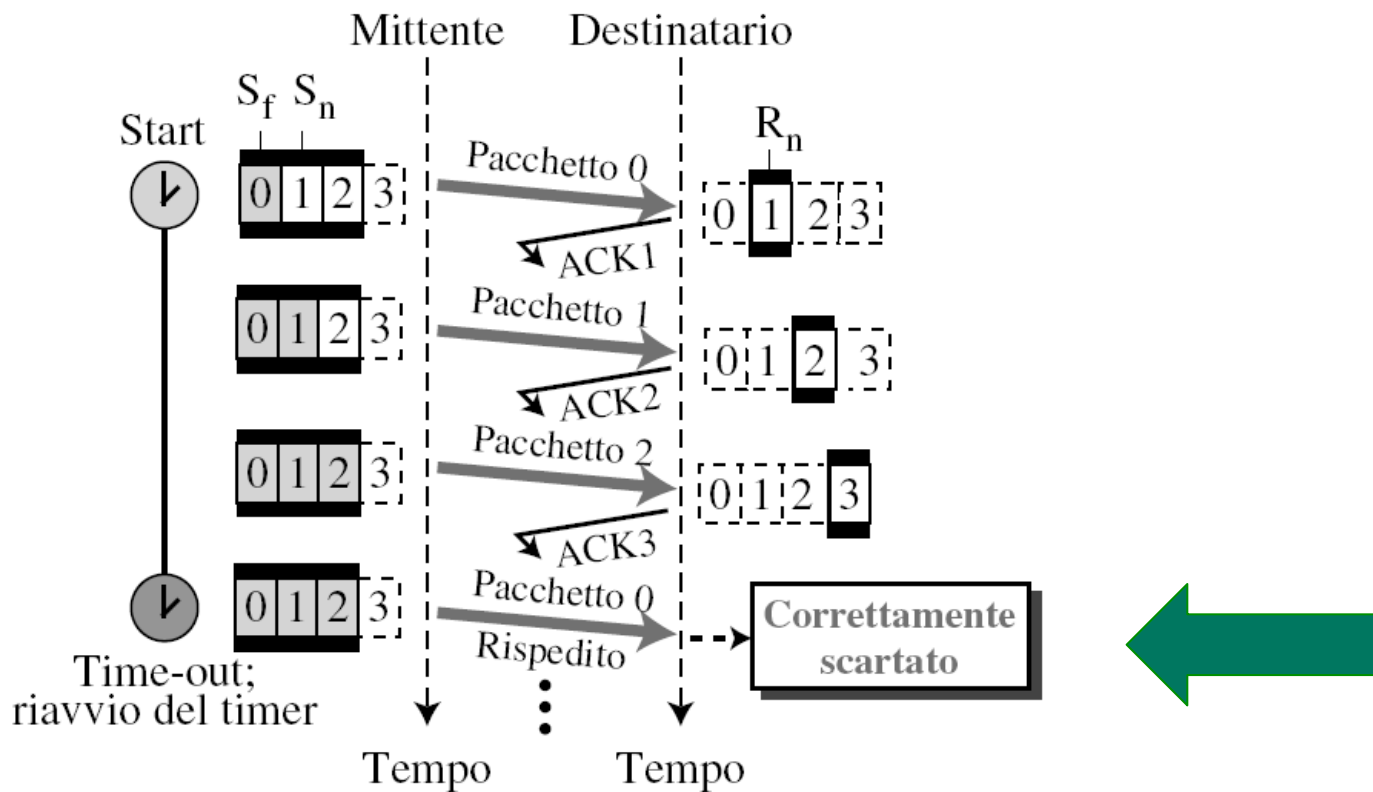
m=2
 $2^2=4$ numeri di sequenza
(0,1,2,3)



Finestra di invio di dimensione = 2^m

Dimensione finestra di invio

- La dimensione deve essere $2^m - 1$

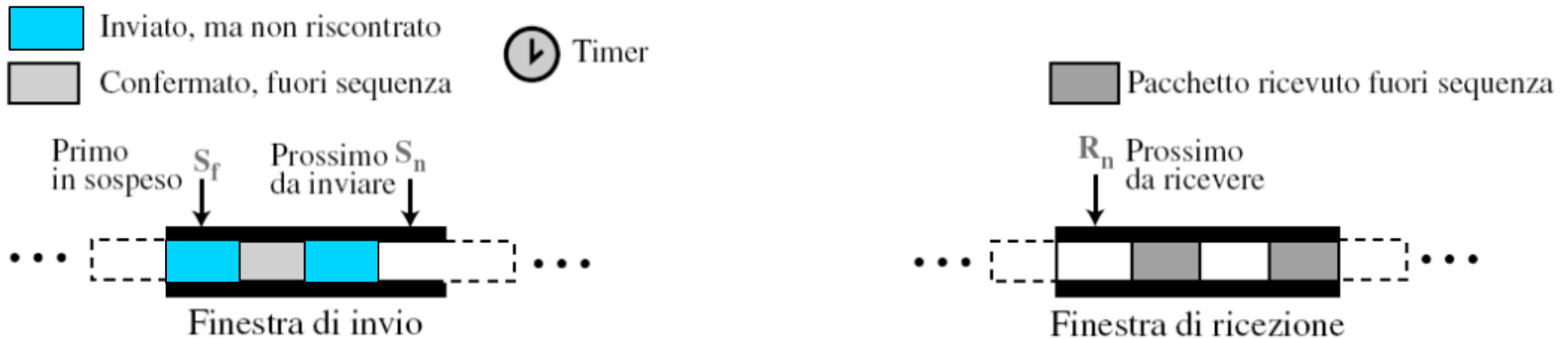
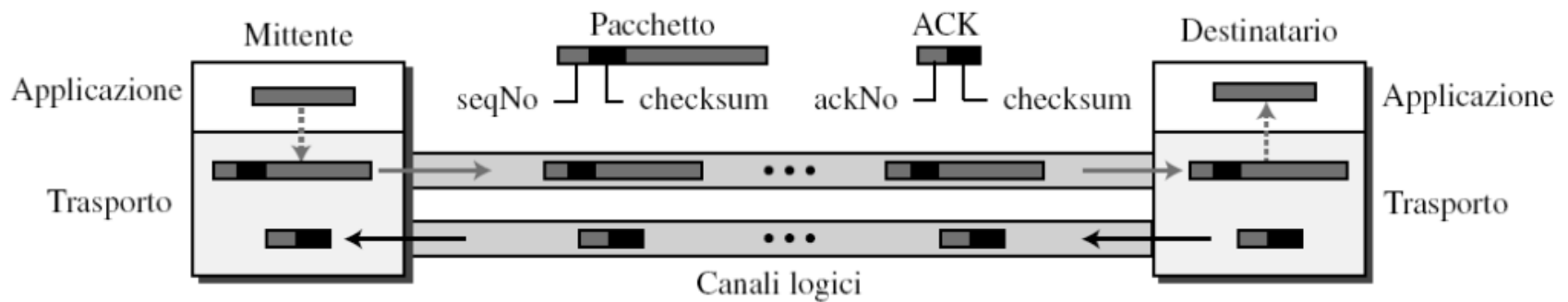


a. Finestra di invio di dimensione $< 2^m$

Ripetizione selettiva

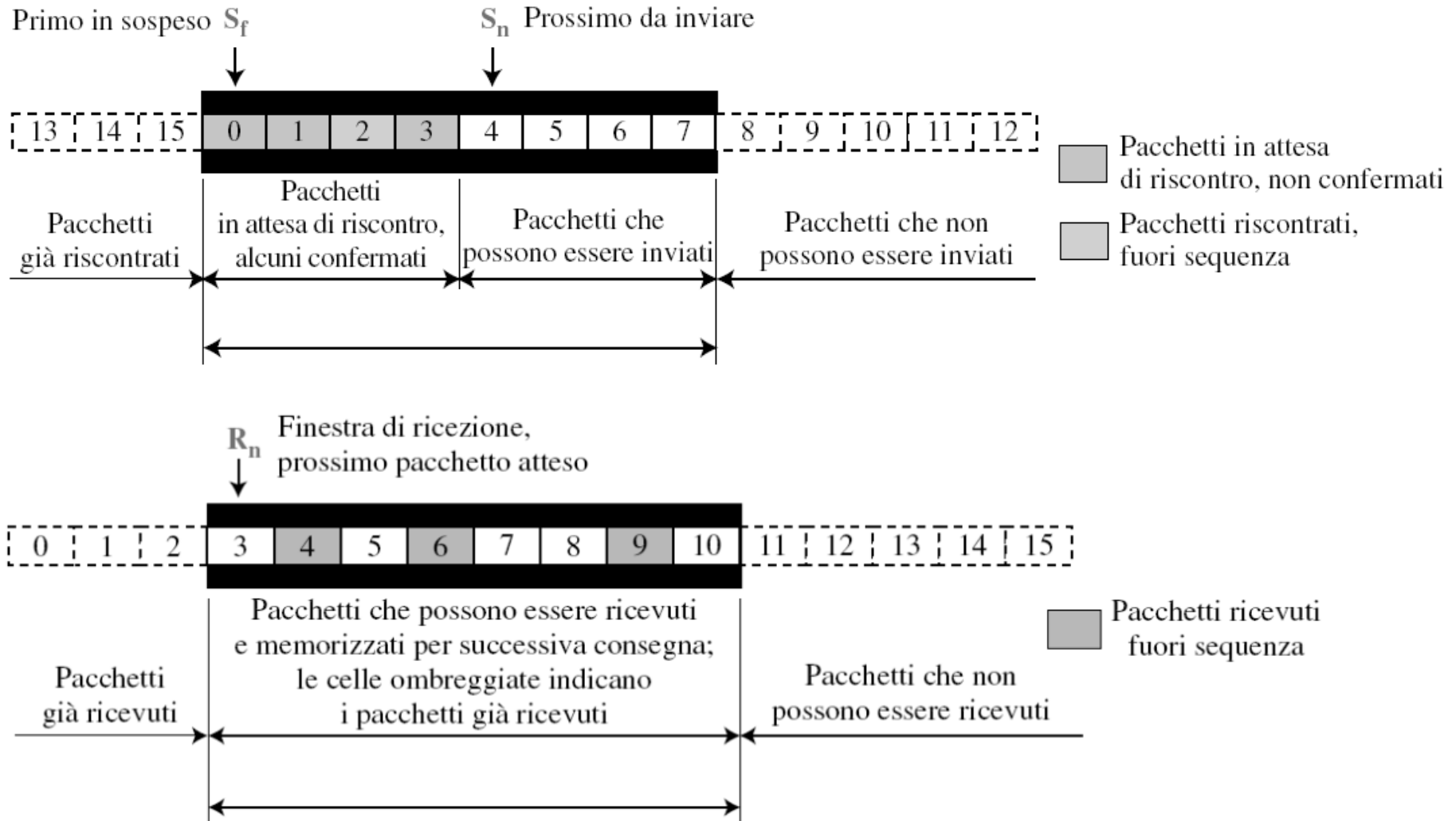
- ❑ In GBN per un solo pacchetto perso si ritrasmettono tutti i successivi già inviati nel pipeline
- ❑ Se in rete c'è congestione, la rispedizione di tutti i pacchetti peggiora la congestione
- ❑ Nella *ripetizione selettiva*, il mittente ritrasmette **soltanto** i pacchetti per i quali non ha ricevuto un ACK
 - timer del mittente per ogni pacchetto non riscontrato
- ❑ Il ricevente invia **riscontri specifici** per tutti i pacchetti ricevuti correttamente (sia in ordine, sia fuori sequenza)
 - buffer dei pacchetti, se necessario, per eventuali consegne in sequenza al livello superiore

Schema generale



Finestra di invio e ricezione per selective repeat

- Finestra di invio e ricezione hanno la stessa dimensione

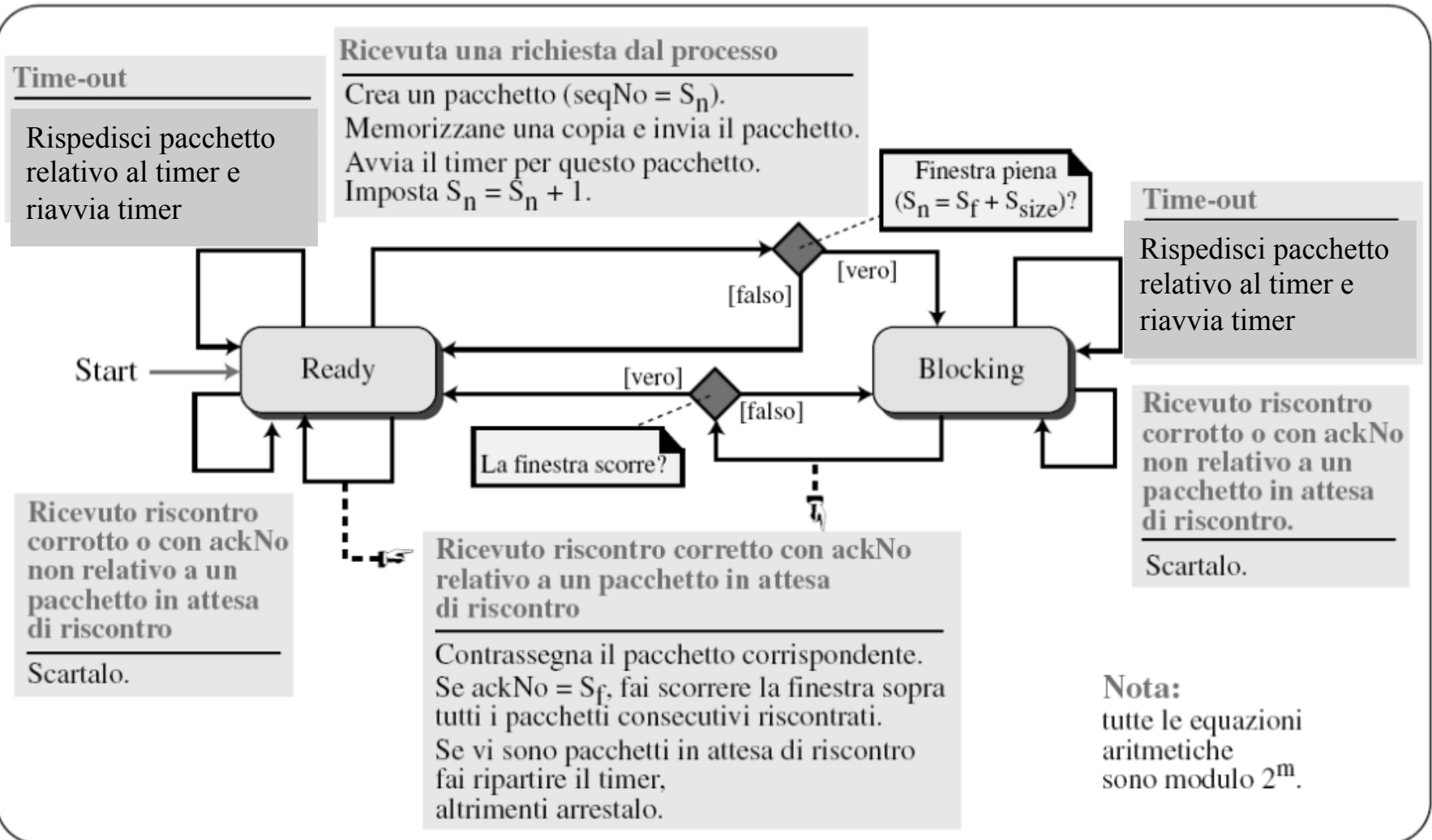


Timer e riscontri

- ❑ *Selective repeat* usa *un timer per ogni pacchetto* in attesa di riscontro
 - Quando scade un timer si rinvia solo il relativo pacchetto
- ❑ *Riscontro individuale* e non cumulativo ma associato al singolo pacchetto: il numero di riscontro indica il numero di sequenza di un pacchetto ricevuto correttamente (non il prossimo atteso)

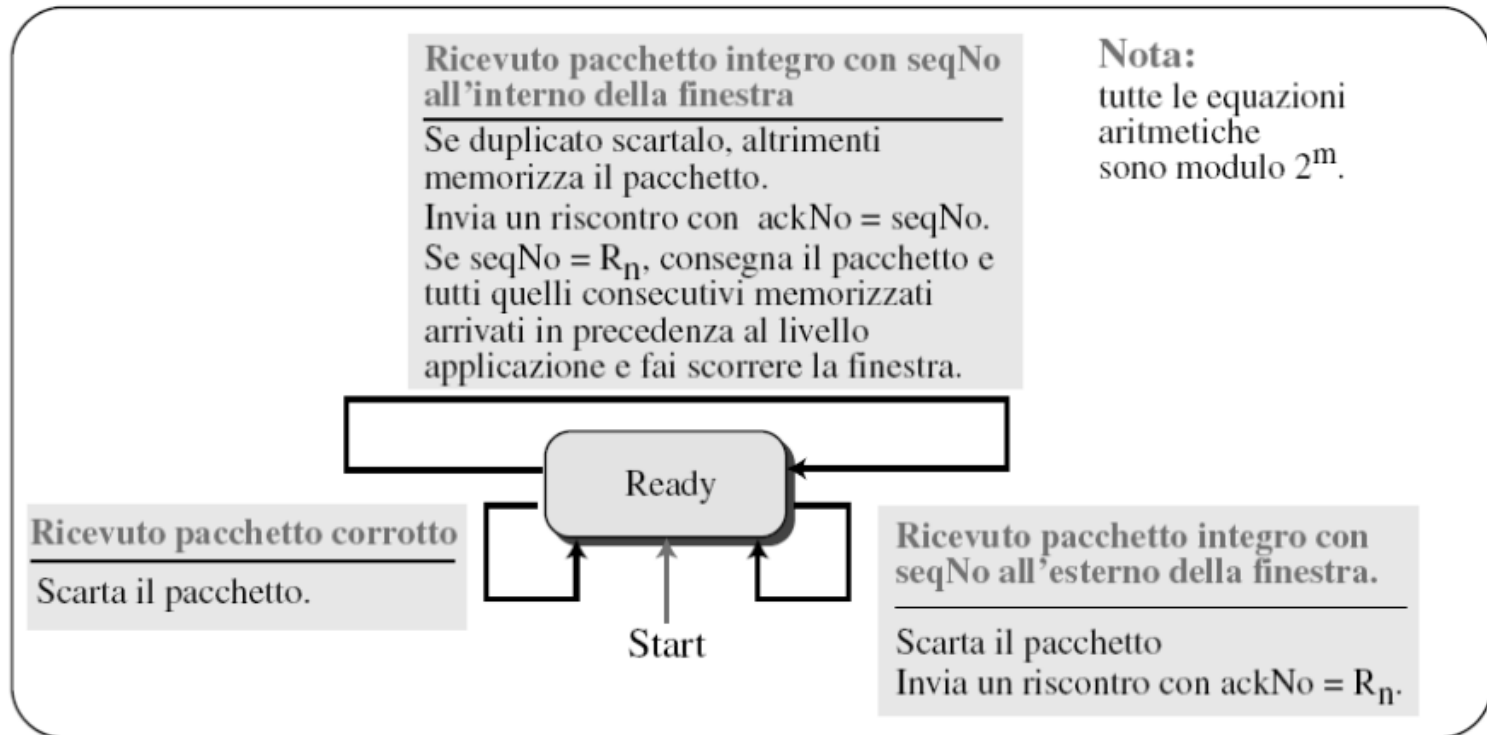
FSM mittente

Mittente

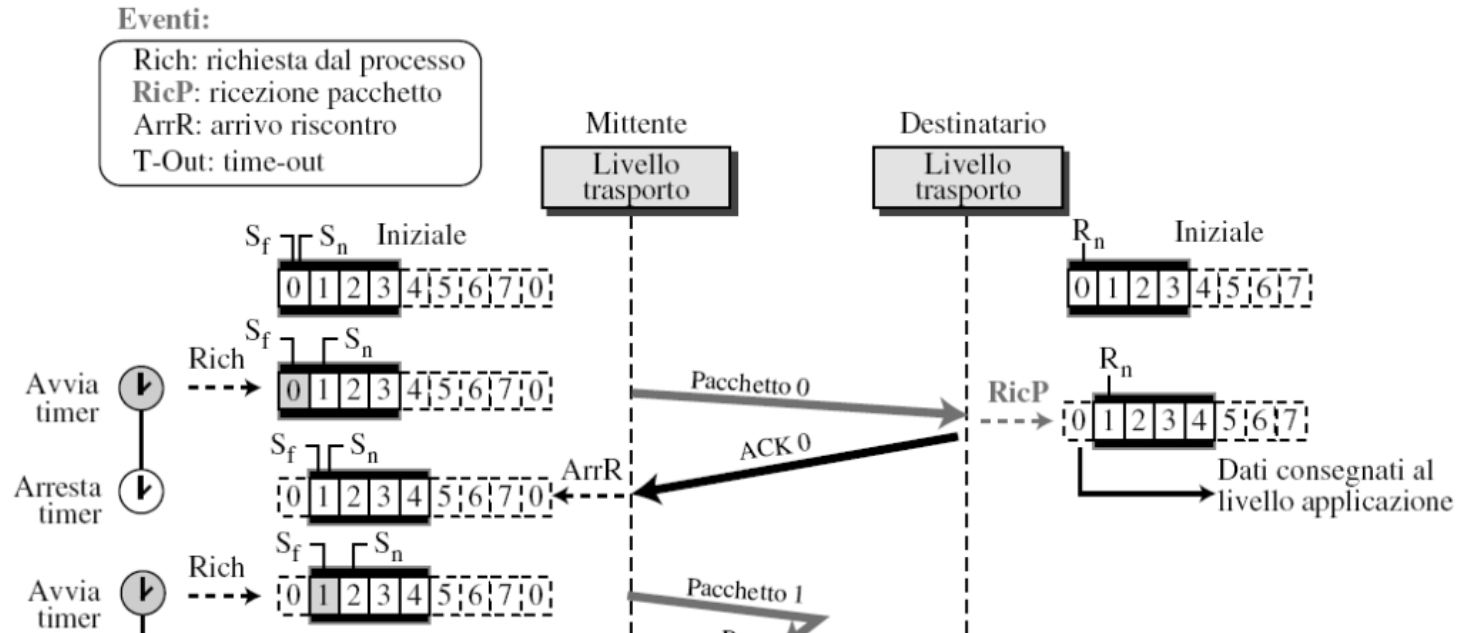


FSM destinatario

Destinatario

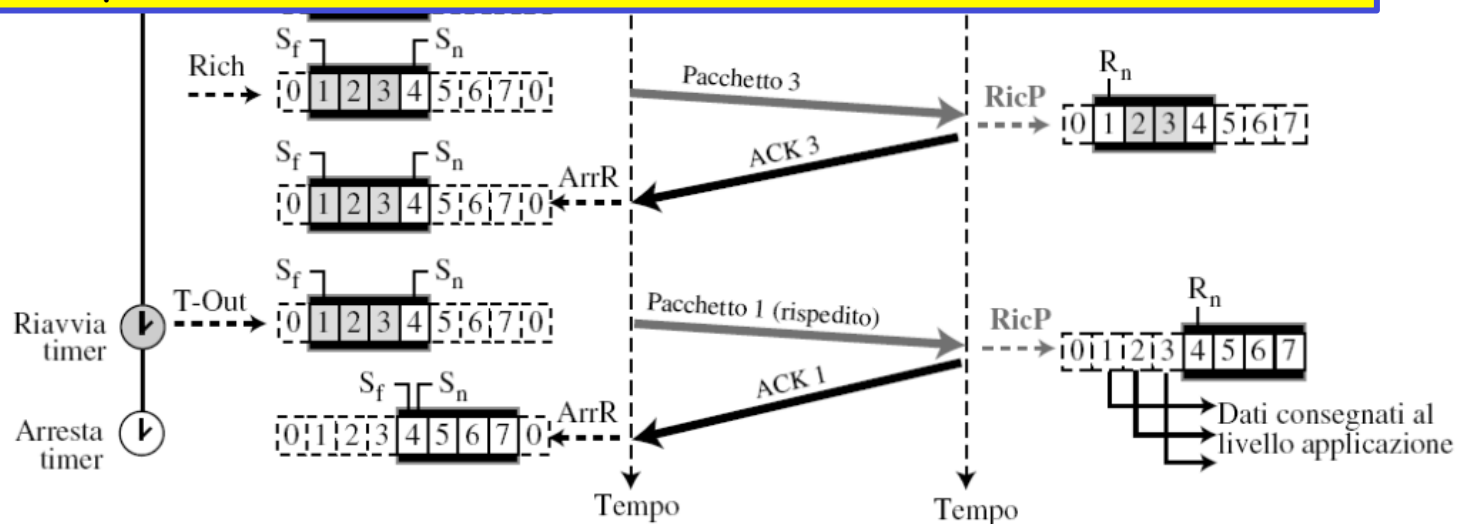


Esempio



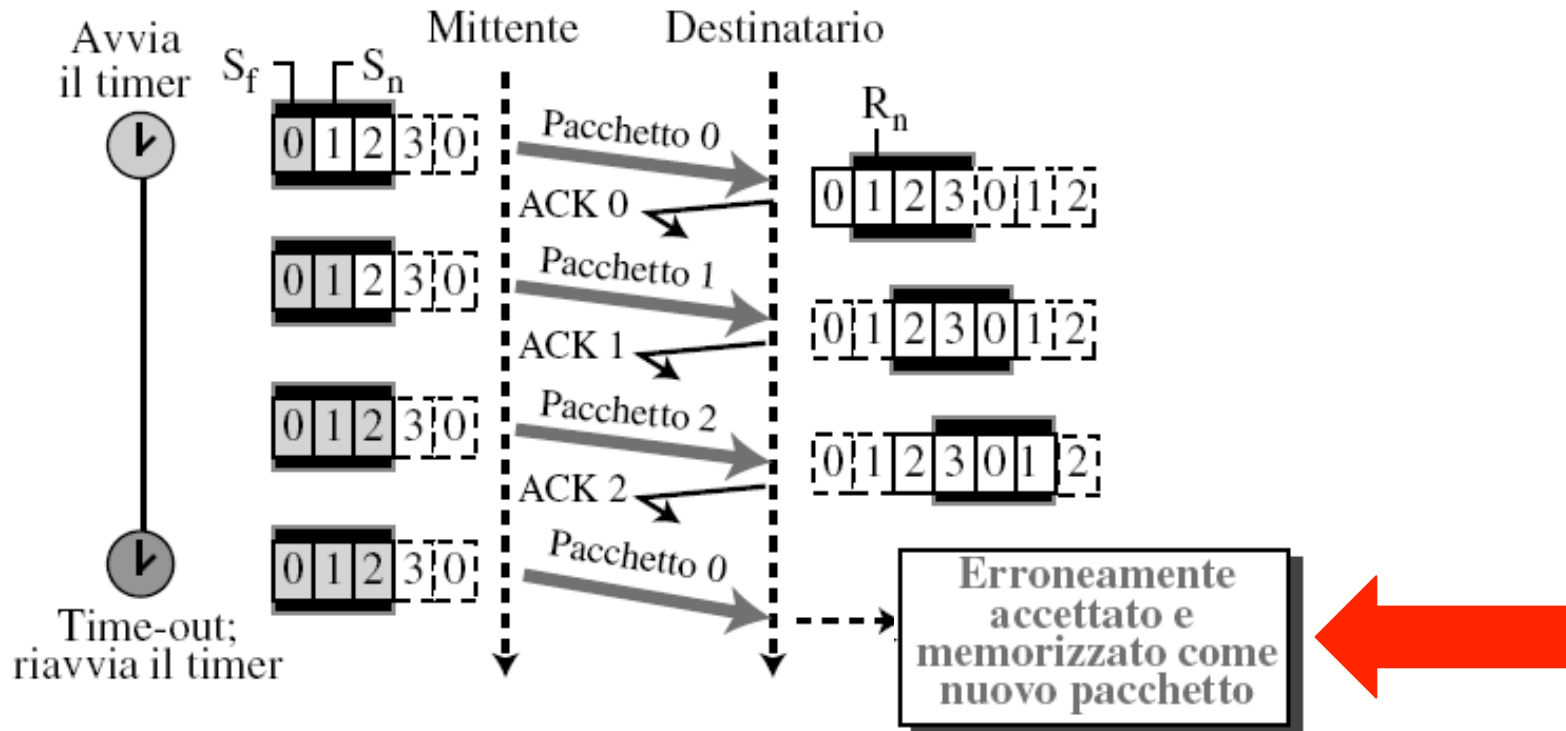
N.B. i pacchetti possono essere consegnati al livello applicazione se:

1. E' stato ricevuto un insieme di pacchetti consecutivi
2. L'insieme deve partire dall'inizio della finestra

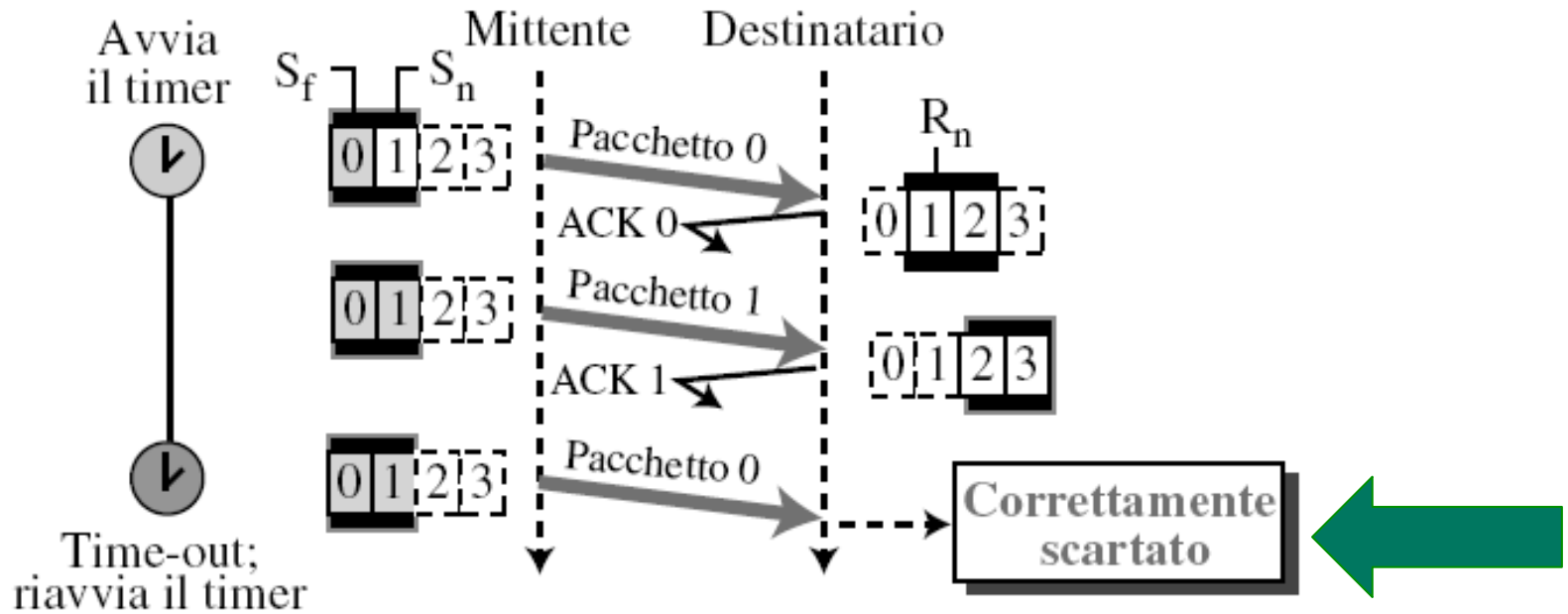


Dimensione delle finestre invio e ricezione

Possiamo usare finestre di dimensione = $2^m - 1$?



Dimensione delle finestre invio e ricezione



a. Finestre di invio e di ricezione di dimensione = $2^m - 1$

Protocolli bidirezionali: piggybacking

- ❑ Abbiamo mostrato meccanismi unidirezionali : pacchetti dati in una direzione e ack nella direzione opposta
- ❑ In realtà entrambi viaggiano nelle due direzioni
- ❑ Per migliorare l'efficienza dei protocolli bidirezionali viene utilizzata la tecnica del *piggybacking*: quando un pacchetto trasporta dati da A a B, può trasportare anche i riscontri relativi ai pacchetti ricevuti da B e viceversa

Riassunto dei meccanismi di trasferimento dati affidabile e loro utilizzo

Meccanismo	Uso	
Checksum	Per gestire errori nel canale	} Gestione inaffidabilità della rete
Acknowledgment	Per gestire errori nel canale	
Numero di sequenza	Ack con errori Perdita pacchetti	
Timeout	Perdita pacchetti	
Finestra scorrevole, pipeling	Maggior utilizzo della rete	→ Miglioramento prestazioni

- ❑ Meccanismi per realizzare un trasferimento dati affidabile in un contesto generale
- ❑ TCP come e quali meccanismi usa per realizzare un trasferimento affidabile?

Esercizio 1

Usando numeri di sequenza a 5 bit, qual è la dimensione massima delle finestre di invio e di ricezione per ciascuno dei meccanismi seguenti?

Stop-and-Wait, Go-Back-N, Selective-Repeat

a. Stop-and-Wait:

Max Send Wsize = 1

Max Receive Wsize = 1

b. Go-Back-N:

Max Send Wsize = $2^5 - 1 = 31$

Max Receive Wsize = 1

c. Selective-Repeat:

Max Send Wsize = $2^{5-1} = 16$

Max Receive Wsize = $2^{5-1} = 16$

Esercizio 2

- In una rete con un valore fisso $m > 1$, è possibile utilizzare entrambi i meccanismi Go-Back-N e Selective-Repeat.
 1. Indicare i vantaggi e gli svantaggi dell'impiego di ciascuno di essi.
 2. Quali altre considerazioni si devono fare per decidere quale meccanismo utilizzare?