Appunti di Fondamenti di Comunicazioni e Internet

Canale 2 · Reti · Prof. Marco Polverini

Capitolo 6: Trasferimento dati affidabile: stop and wait, Go-Back-N e Selective Repeat

Autore: Tommaso C.

Appunti tratti dalle lezioni in aula e dal materiale del corso

---

Argomenti: Reliable data transfer · End-to-end e hop by hop
Stop and wait: riscontri, timeout, numeri di sequenza · Efficienza con e senza errori
Pipeline · Go-Back-N: finestra scorrevole, numeri a m bit, piggybacking, timeout
Selective Repeat · Confronto dei tre protocolli

Indice

1 Il trasferimento dati affidabile **1**

2 Dove si realizza: end-to-end e hop by hop **1**

2.1 Perché servono tutti e due 2

3 Il protocollo stop and wait **2**

3.1 Codice, riscontro e timeout 2

3.2 Duplicati e numeri di sequenza 3

3.3 Numeri 0 e 1 5

3.4 Macchina a stati finiti 5

4 Efficienza dello stop and wait **6**

4.1 Efficienza con errori 8

5 Protocolli con pipeline **10**

6 Il protocollo Go-Back-N **11**

6.1 Finestra scorrevole 11

6.2 Perdite e ritrasmissione 12

6.3 Riscontri cumulativi 13

6.4 Frame fuori ordine 13

6.5 Quale finestra 14

6.6 Numeri di sequenza a m bit 14

6.7 Piggybacking 15

6.8 Durata del timeout 15

7 Il protocollo Selective Repeat **15**

7.1 Il limite del Go-Back-N 16

7.2 Confronto dei tre protocolli 17

8 Esercizi **19**

9 Soluzioni **23**

1 Il trasferimento dati affidabile

Quando il ricevitore scopre che un frame ha errori (Capitolo 5) ha tre strade: scartarlo, correggerlo, se il codice lo permette, oppure recuperarlo con una procedura di **trasferimento dati affidabile**. Correggere può correggere male, e scartare perde il dato.

Definizione – Trasferimento dati affidabile

Il **trasferimento dati affidabile** (*reliable data transfer*, **RDT**) consegna i dati **senza errori**, **una sola volta** e **nell'ordine di invio**, anche se il canale è inaffidabile.

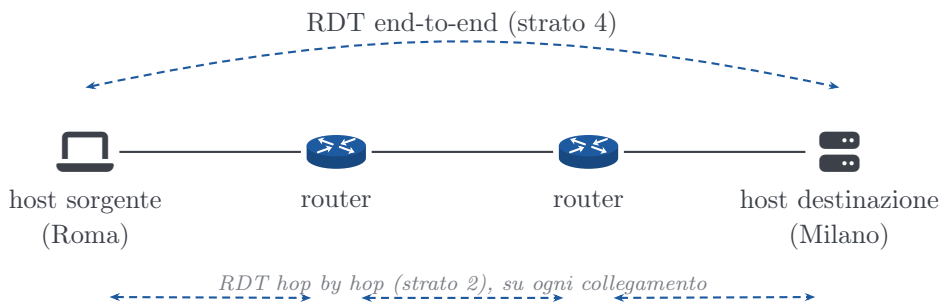
Un canale inaffidabile può fare quattro cose, le stesse del servizio best effort (Capitolo 4):

Evento	Che cosa succede	Esempio
errore	arrivano bit diversi da quelli trasmessi	trasmesso 110, ricevuto 100
perdita	il messaggio non arriva mai	un guasto lungo il collegamento
duplicato	lo stesso messaggio arriva due volte	M e subito dopo ancora M
fuori sequenza	l'ordine di arrivo è invertito	M_2, generato dopo, arriva prima di M_1

2 Dove si realizza: end-to-end e hop by hop

Definizione – RDT end-to-end e hop by hop

- L'RDT **end-to-end** è realizzato solo dai sistemi terminali, sorgente e destinazione, nello **strato di trasporto** (strato 4). Lo fa TCP.
- L'RDT **hop by hop** (salto dopo salto) è realizzato su ogni collegamento, dallo **strato di collegamento** (strato 2).



2.1 Perché servono tutti e due

Il progettista di un'applicazione non sa quali collegamenti attraverseranno i suoi dati: sa solo di che servizi ha bisogno. Se gli serve l'RDT lo chiede allo strato di trasporto, quindi lo ottiene end-to-end. Per questo HTTP è trasportato da TCP.

Ripetere l'RDT a più livelli costa: è lavoro in più e rallenta il sistema. Conviene farlo anche sui collegamenti solo dove serve.

	End-to-end (strato 4)	Hop by hop (strato 2)
Chi lo decide	chi progetta l'applicazione, che lo chiede al trasporto	la tecnologia del collegamento: Ethernet non lo offre, il 5G sì
Vantaggio	vale su qualunque percorso, anche se qualche collegamento non offre l'RDT	recupera subito, sul primo salto, dove gli errori sono più probabili
Svantaggio	il recupero è lento: serve un tempo di andata e ritorno fra i terminali, decine di millisecondi fra Roma e New York	ripetere il servizio su ogni collegamento appesantisce la rete

Sui collegamenti cablati, come la fibra, gli errori binari sono molto rari e l'RDT non serve. Sui collegamenti senza fili o via satellite sono frequenti.

3 Il protocollo stop and wait

Definizione – ARQ e stop and wait

Gli **ARQ** (*Automatic Repeat Request*) sono una famiglia di protocolli che realizza l'RDT sopra un collegamento inaffidabile. Il più semplice è lo **stop and wait**: il trasmettitore A invia un frame e **si ferma ad aspettare** il riscontro, e solo dopo invia il successivo. Il ricevitore è B.

3.1 Codice, riscontro e timeout

1. A non trasmette M ma una **parola di codice**, con bit di controllo (Capitolo 5): B può così rivelare gli errori.
2. Se la stringa ricevuta è una parola di codice, B **accetta** il frame.
3. A non può sapere che la ricezione è riuscita, se B non glielo dice. B invia un **riscontro** (**ACK**, *acknowledgement*), un messaggio **positivo**: arriva solo quando le cose vanno bene.

L'ACK è un frame a sé: ha un'intestazione e un codice che ne protegge dagli errori, ma nessun dato dell'utente.

Se il frame si perde, B non riceve nulla e non risponde, e A non vede mai arrivare un ACK: i due restano in stallo.

Definizione – Timeout di ritrasmissione

A ogni frame trasmesso A avvia un **timeout**, detto **RTO** (*Retransmission Timeout*). Se scade prima dell'arrivo dell'ACK, A **ritrasmette** il frame.

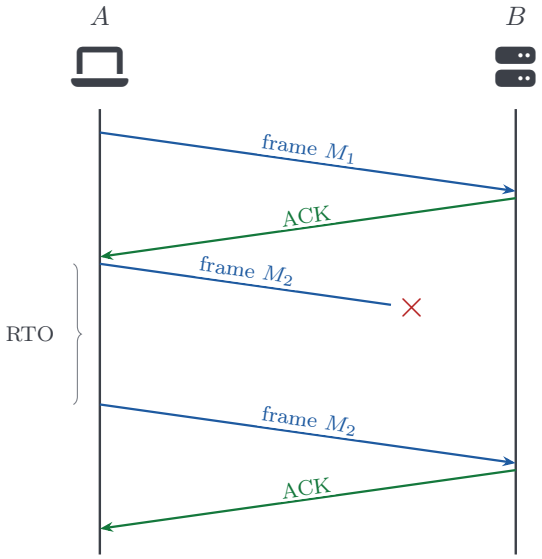


Figura 1: Un frame perso. B non riceve M_2 e non risponde. Allo scadere dell'RTO A ritrasmette M_2.

3.2 Duplicati e numeri di sequenza

L'RTO può scadere anche se il frame è arrivato: il frame, o il suo ACK, è molto in ritardo. A ritrasmette un frame che B ha già accettato, e B, che non può saperlo, accetta anche la copia. Il ricevitore consegna così lo stesso messaggio due volte.

Definizione – Numero di sequenza

Un **numero di sequenza** è un identificativo univoco scritto nell'intestazione del frame, per esempio un contatore che aumenta di uno a ogni frame. Una ritrasmissione porta lo stesso numero del frame originale, quindi B riconosce il **duplicato**: lo **scarta**, e invia di nuovo l'ACK, perché quello che aveva mandato non ha sbloccato A.

Anche gli ACK portano un numero.

Definizione – ACK numerato

L'ACK porta il numero di sequenza del **prossimo frame che B si aspetta** (*expected sequence number*). Dopo il frame 0 l'ACK porta 1. A **ignora** un ACK che non è quello che aspetta.

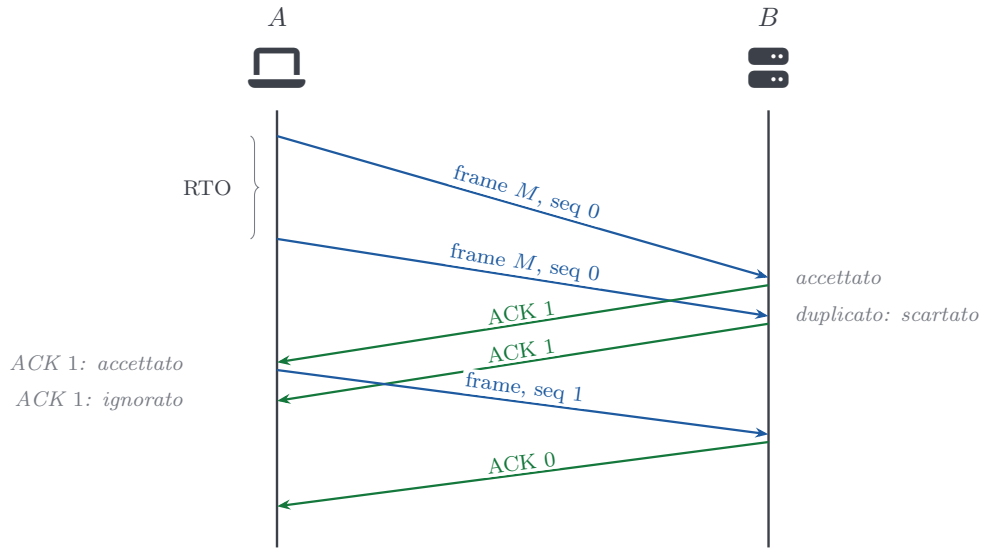


Figura 2: Il primo frame M è molto in ritardo e l'RTO scade: A ritrasmette M con lo stesso numero 0. B accetta il primo che arriva, scarta il secondo come duplicato e risponde a tutti e due con ACK 1. Il primo ACK sblocca A, che passa al frame 1 e ignora il secondo ACK 1, perché aspetta l'ACK 0.

Osservazione – Ricevere e consegnare

Ricevere un frame e consegnarlo allo strato superiore sono cose diverse: B riceve il duplicato, ma non ne consegna il contenuto.

Proposizione – I meccanismi dello stop and wait

Lo stop and wait si regge su cinque meccanismi:

- un **codice di rivelazione** degli errori, per accorgersi dei frame errati;
- i **riscontri positivi** (ACK), per sapere che la ricezione è riuscita;
- il **timeout** (RTO), per uscire dallo stallo quando frame o ACK si perdono;
- la **ritrasmissione**, per recuperare i frame persi o errati;
- i **numeri di sequenza**, per riconoscere duplicati e frame fuori sequenza.

3.3 Numeri 0 e 1

Definizione – Numerazione a un bit

Nello stop and wait basta usare i numeri di sequenza 0 e 1, alternati. C'è sempre al più un frame non riscontrato in giro, perché dopo ogni frame A si ferma: ritrasmette solo se scade l'RTO, e riprende solo con l'ACK. Se B riceve 0, poi 1 e poi di nuovo 0, sa che la catena è ripartita e che quel 0 è un frame nuovo.

Il protocollo non è però infallibile. Un frame può restare in rete molto più a lungo del previsto, e arrivare quando B si aspetta di nuovo lo stesso numero.

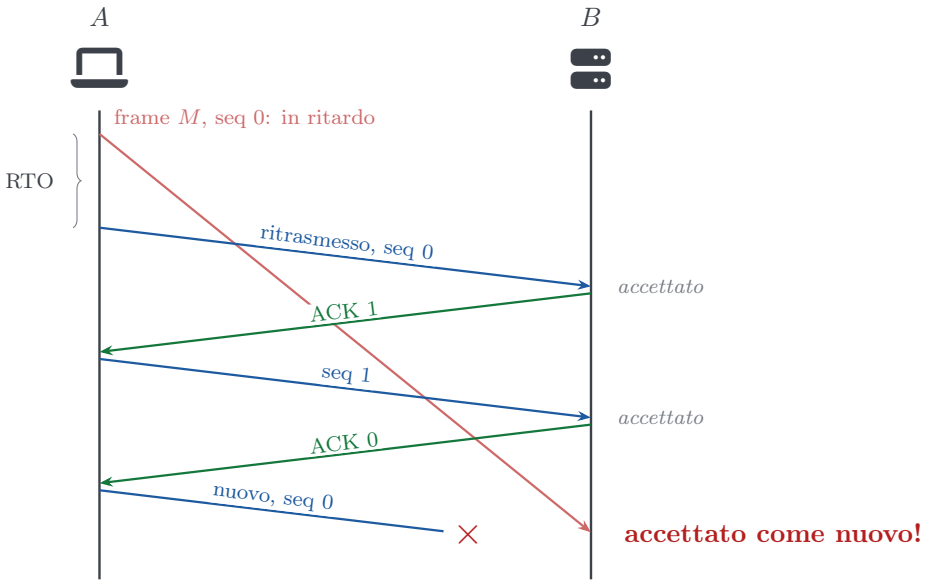


Figura 3: Il frame 0 originale resta in rete. La ritrasmissione e il frame 1 vanno a buon fine, e B aspetta di nuovo il numero 0. Il nuovo frame 0 si perde, e proprio allora arriva il vecchio: B lo accetta e lo consegna come se fosse quello nuovo.

Con più numeri di sequenza il rischio si riduce, ma si paga: il ricevitore ha bisogno di più memoria, e i numeri sono bit dell'intestazione, quindi overhead. Più piccolo è il campo, meno overhead.

3.4 Macchina a stati finiti

Definizione – Stati dello stop and wait

Lo stato del sistema è la coppia $(S_{\text{last}}, R_{\text{next}})$: il numero del frame che A sta trasmettendo e il numero che B si aspetta. Si parte da $(0, 0)$. Senza errori il sistema percorre quattro stati: ogni frame corretto e atteso fa passare B al numero successivo, ogni ACK atteso fa passare A al frame successivo.

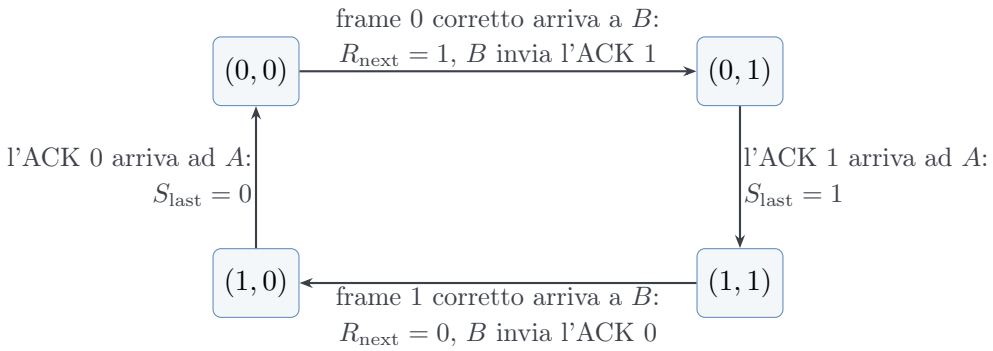


Figura 4: Gli stati $(S_{\text{last}}, R_{\text{next}})$ senza errori.

Negli altri casi lo stato non cambia. Se a B arriva un frame senza errori ma con un numero diverso da quello atteso, un duplicato, B lo scarta e invia l'ACK con R_{next}, cioè dice quale frame aspetta. Un frame con errori B lo scarta e basta, senza rispondere. Se ad A arriva un ACK diverso da quello atteso, A lo ignora.

4 Efficienza dello stop and wait

Definizione – Efficienza

L'**efficienza** η è il rapporto fra il **tempo utile** e il **tempo totale** di un frame. Il tempo utile è il tempo che A passa a trasmettere il frame, $T_{\text{TX}} = L/C$. Il tempo totale va dal primo istante di trasmissione alla ricezione dell'ACK.

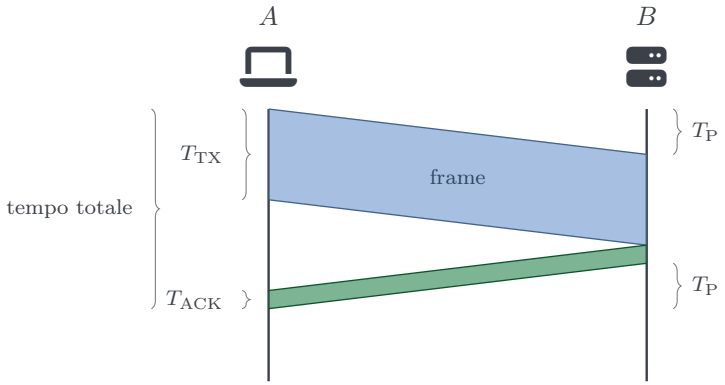


Figura 5: Un ciclo dello stop and wait. Il tempo utile è T_{TX}, quello totale va fino all'arrivo dell'ACK.

Il tempo totale comprende la trasmissione del frame, due volte la propagazione e la trasmissione dell'ACK. L'ACK è lungo L_H bit, con $T_{\text{ACK}} = L_H/C$: ha solo l'intestazione e il CRC.

Proposizione – Efficienza dello stop and wait

$$\eta = \frac{L/C}{L/C + 2\,T_{\mathrm{P}} + L_H/C} = \frac{1}{1 + \dfrac{2\,C\,T_{\mathrm{P}}}{L} + \dfrac{L_H}{L}}.$$

Con $L_H \ll L$ il termine dell'ACK si trascura. Con $\mathrm{RTT} = 2\,T_{\mathrm{P}}$,

$$\boxed{\eta \simeq \frac{1}{1 + C\,\mathrm{RTT}/L}}.$$

L'RTT di questa formula è solo $2\,T_{\mathrm{P}}$, senza i tempi di trasmissione: non è l'RTT del Capitolo 3.

L'ACK, inoltre, spesso non viaggia da solo: se la comunicazione è bidirezionale, B lo inserisce in un frame di dati diretto verso A (*piggybacking*, Sezione 6), e il suo costo scompare.

Definizione – Prodotto banda-ritardo

Il **prodotto banda-ritardo** (*bandwidth-delay product*, BDP) è

$$\mathrm{BDP} = C \cdot \mathrm{RTT}.$$

Si misura in bit: sono i bit che possono stare in transito nel collegamento, come l'acqua in un tubo di sezione C e lunghezza RTT.

Lo stop and wait mette nel collegamento un solo frame alla volta. Se il BDP è piccolo rispetto a L il frame riempie quasi il tubo e l'efficienza è alta; se è grande il tubo resta quasi vuoto. Il BDP sta al denominatore di η.

Esempio – Efficienza al variare del ritardo e della capacità

Frame di $L = 10\,000$ bit e ACK di $L_H = 200$ bit. Con $v = 2 \times 10^8$ m/s (Capitolo 3) i ritardi corrispondono a collegamenti lunghi da $d = 100$ km a $d = 100\,000$ km. Per ogni RTT si calcola BDP $= C \cdot \mathrm{RTT}$ e $\eta = 1/(1 + \mathrm{BDP}/L + L_H/L)$:

		$C = 1$ Mbit/s		$C = 1$ Gbit/s	
d (km)	RTT	BDP (bit)	η	BDP (bit)	η
100	1 ms	10^3	89,3 %	10^6	0,990 %
1000	10 ms	10^4	49,5 %	10^7	0,0999 %
10 000	100 ms	10^5	9,07 %	10^8	0,0100 %
100 000	1 s	10^6	0,990 %	10^9	0,001 00 %

A 1 Gbit/s, la capacità d'accesso di una fibra domestica, l'efficienza è già dell'1% con $d = 100$ km: si usa un centesimo della capacità.

4.1 Efficienza con errori

Finora ogni frame arrivava corretto. Se il canale sbaglia, alcuni frame arrivano con errori: B li scarta e A li ritrasmette, e ogni ritrasmissione costa un ciclo in più.

Definizione – Probabilità d'errore sul frame

P_f è la probabilità che un frame arrivi con errori; $1 - P_f$ è la probabilità che arrivi corretto. Se i bit sbagliano in modo indipendente con probabilità BER, un frame di L bit è corretto se lo sono tutti i suoi bit:

$$1 - P_f = (1 - \mathrm{BER})^L \simeq e^{-L \cdot \mathrm{BER}},$$

e l'approssimazione vale per L grande e BER piccolo.

Sia N il numero di trasmissioni che servono per far arrivare corretto un frame. Le trasmissioni sono indipendenti: $N = n$ se le prime $n - 1$ sbagliano e l'n-esima no,

$$P(N = n) = P_f^{n-1}\,(1 - P_f), \qquad n = 1, 2, \ldots$$

Proposizione – Numero medio di trasmissioni

Il numero medio di trasmissioni di un frame è

$$E[N] = \sum_{n=1}^{\infty} n\, P_f^{n-1}(1 - P_f) = \boxed{\frac{1}{1 - P_f}}.$$

Dimostrazione

Per $|x| < 1$ la serie geometrica vale $\sum_{n=0}^{\infty} x^n = 1/(1-x)$. Derivando termine a termine,

$$\sum_{n=1}^{\infty} n\, x^{n-1} = \frac{1}{(1-x)^2}.$$

Con $x = P_f$ si ottiene $E[N] = (1-P_f) \cdot \dfrac{1}{(1-P_f)^2} = \dfrac{1}{1-P_f}$. □

Se l'RTO è uguale al ciclo T_{TOT} e gli ACK arrivano sempre, ogni trasmissione dura un ciclo, e il tempo medio per un frame è $T_{\text{TOT}}/(1-P_f)$. Il tempo utile resta L/C, quindi

$$\boxed{\eta = \frac{L/C}{T_{\text{TOT}}/(1-P_f)} = (1-P_f)\,\eta_{\text{SW}}},$$

dove η_{SW} è l'efficienza senza errori della Sezione 4.

Per segnalare l'errore B può anche rispondere con un **riscontro negativo** (NAK), che chiede subito la ritrasmissione senza aspettare l'RTO. Nello stop and wait il NAK porta lo stesso numero dell'ultimo ACK: «aspetto ancora quel frame». L'Esercizio 2 calcola la capacità minima del canale di ritorno con questa regola.

Esempio – Efficienza dello stop and wait e BER

Con i dati della prima riga dell'esempio precedente ($L = 10\,000$ bit, $L_H = 200$ bit, $C = 1$ Mbit/s, RTT $= 1$ ms, $\eta_{\text{SW}} = 89{,}3\,\%$):

BER	0	10^{-6}	10^{-5}	10^{-4}
$1-P_f$	1	0,990	0,905	0,368
η	89,3 %	88,4 %	80,8 %	32,8 %

L'efficienza crolla quando $L \cdot \text{BER}$ si avvicina a 1: con $L \cdot \text{BER} = 1$ arriva corretto poco più di un frame su tre.

Lo stop and wait mostra i principi del recupero degli errori, ma funziona male: servono protocolli con prestazioni più alte.

L'RTO non può durare meno del tempo totale: prima, l'ACK non può essere arrivato. Ogni frame perso, o ACK perso, costa quindi almeno un RTO in più: se il tempo totale di un ciclo è $T_{\text{TOT}} = T_{\text{TX}} + 2\,T_{\text{P}} + T_{\text{ACK}}$ e si ritrasmette k volte, n frame richiedono

$$n\,T_{\text{TOT}} + k\,\text{RTO}.$$

5 Protocolli con pipeline

Lo stop and wait si ferma dopo ogni frame, e per questo tempo utile e totale distano. Un frame molto lungo non risolve: la lunghezza non la decide chi trasmette, e un frame lungo è più vulnerabile agli errori binari. Si può invece **non fermarsi**.

Definizione – Protocolli con pipeline

Un protocollo **con pipeline** ammette sul collegamento più frame **in volo**, cioè trasmessi e non ancora riscontrati, ciascuno con il proprio numero di sequenza. Dopo avere trasmesso W frame il trasmettitore aspetta i riscontri; W è un parametro che si può impostare.

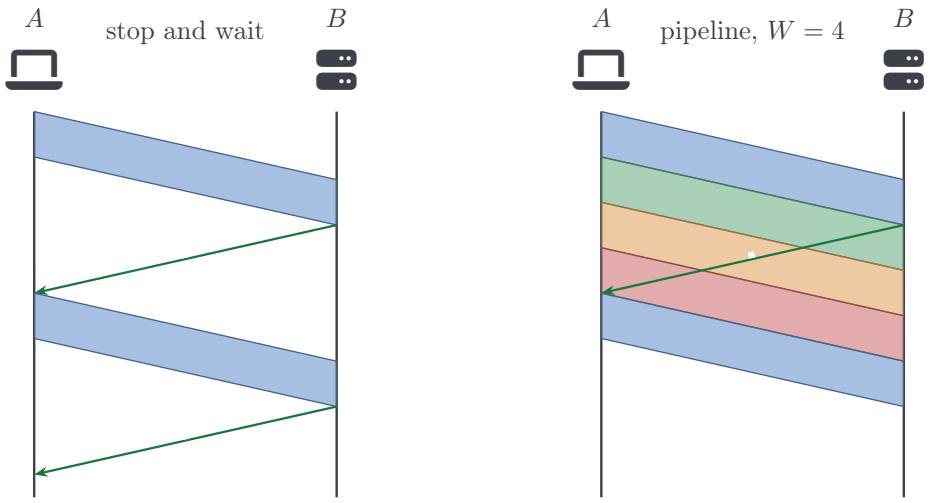


Figura 6: Stop and wait: un frame alla volta nel collegamento. Pipeline con $W = 4$: i frame si seguono senza pause, e il primo ACK arriva quando l'ultimo dei quattro finisce di essere trasmesso.

Proposizione – Efficienza con pipeline

Con W frame per ciclo l'efficienza è W volte quella dello stop and wait, ma non supera 1. Trascurando l'ACK,

$$\eta = \min\left(1,\ \frac{W\,L/C}{L/C + \mathrm{RTT}}\right) = \min\left(1,\ \frac{W}{1 + C\,\mathrm{RTT}/L}\right).$$

Il numero minimo di frame che dà $\eta = 1$, cioè una trasmissione senza interruzioni, è

$$\boxed{W_{\min} = \left\lceil 1 + \frac{C\,\mathrm{RTT}}{L} \right\rceil}$$

Se si tiene conto anche dell'ACK basta contare il ciclo completo: la trasmissione è senza interruzioni con $W \geq T_{\text{TOT}}/T_{\text{TX}}$, e

$$W_{\min} = \left\lceil \frac{T_{\text{TX}} + 2\,T_{\text{P}} + T_{\text{ACK}}}{T_{\text{TX}}} \right\rceil = \left\lceil \frac{1}{\eta_{\text{SW}}} \right\rceil,$$

dove η_{SW} è l'efficienza dello stop and wait. Si applica agli Esercizi 5 e 6.

6 Il protocollo Go-Back-N

6.1 Finestra scorrevole

Nel **Go-Back-N** i frame in volo sono più di uno, quindi i numeri di sequenza non sono più soltanto 0 e 1: il campo dell'intestazione usa più bit.

Definizione – Finestra scorrevole

La **finestra** è una memoria di A lunga W frame. Comincia da un numero detto **base** e contiene i numeri di sequenza da base a base $+\, W - 1$, quelli che A è autorizzato a inviare:

- i numeri di frame già inviati e non ancora riscontrati;
- i numeri ancora liberi, che A può usare per frame nuovi.

Quando arriva un ACK, la base diventa il numero scritto nell'ACK e la finestra **scorre** in avanti (*sliding window*). Se la finestra è piena di frame inviati, A si ferma.

L'ACK porta l'**expected sequence number** (ESN), cioè il numero del prossimo frame atteso: è una variabile di stato di B, che parte da 0 e aumenta di uno a ogni frame accettato. Dopo il frame 0 l'ACK porta 1, e la base diventa 1.

Con $W \geq W_{\min}$ (Sezione 5) l'ultimo bit dell'ultimo frame della finestra parte subito prima che arrivi l'ACK del frame più vecchio, e il collegamento resta sempre pieno: $\eta = 1$.

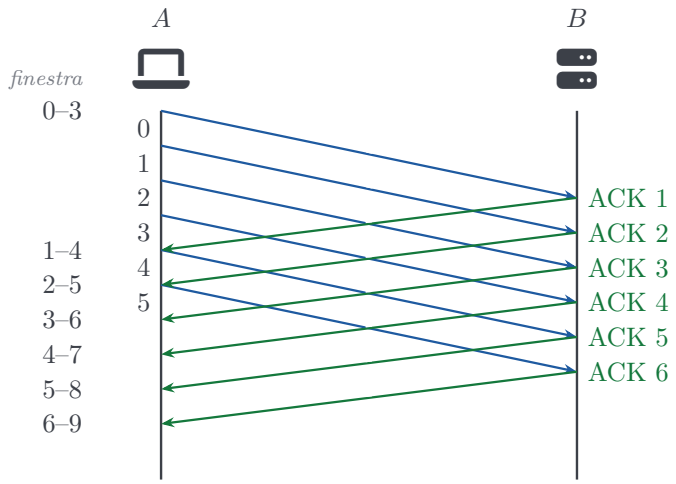


Figura 7: Go-Back-N con $W = 4$. A invia i frame da 0 a 3 e si ferma. L'ACK 1 arriva proprio quando finisce il frame 3: la base passa a 1 e A invia il 4, e così via, senza pause.

6.2 Perdite e ritrasmissione

Definizione – Go-Back-N: regole di A e di B

Ricevitore B: accetta solo il frame corretto con numero uguale all'ESN, lo consegna, aumenta l'ESN di uno e invia l'ACK con il nuovo ESN. Un frame **fuori sequenza** lo scarta e rinvia l'ultimo ACK, cioè l'ESN. Un frame con errori lo scarta senza rispondere.

Trasmettitore A: tiene un solo timer, l'RTO, per il **frame più vecchio non ancora riscontrato**. Quando l'ACK di quel frame arriva, ferma il timer e ne avvia uno per il nuovo frame più vecchio. Se l'RTO scade, A **torna indietro** e ritrasmette tutti i frame della finestra ancora in attesa di riscontro.

Nella Figura 8 la finestra è lunga 3 e il frame 1 si perde:

1. il frame 0 arriva e B risponde con l'ACK 1; la base di A passa a 1, ma la finestra non si svuota, perché 1 e 2 aspettano il riscontro;
2. il frame 2 arriva fuori sequenza: B lo scarta e ripete l'ACK 1, e la finestra di A non scorre;
3. l'RTO del frame 1, avviato all'arrivo dell'ACK 1, scade: A ritrasmette 1 e 2.

Il Go-Back-N paga così l'efficienza: dopo una perdita ritrasmette anche frame che erano arrivati, come il 2.

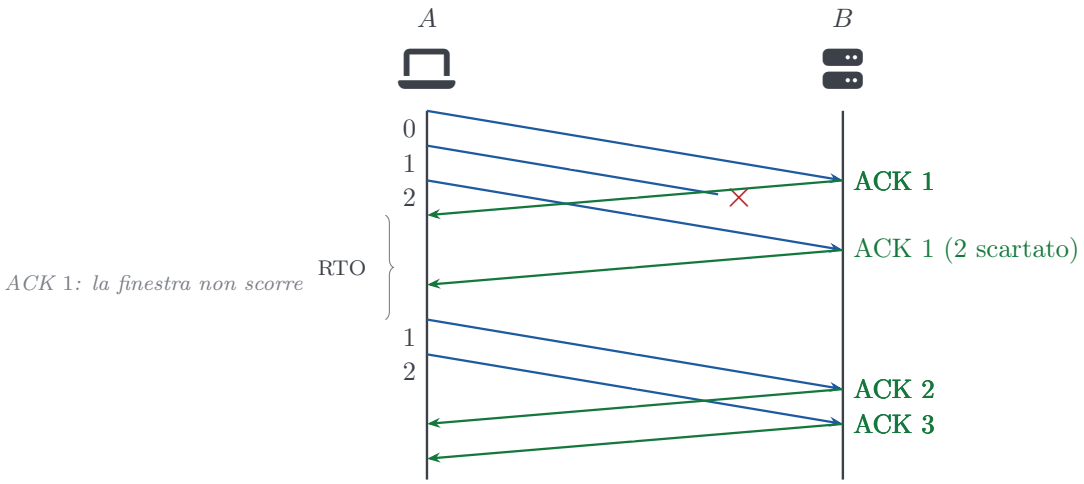


Figura 8: Go-Back-N con $W = 3$ e il frame 1 perso. B scarta il frame 2, fuori sequenza, e ripete l'ACK 1. Allo scadere dell'RTO A torna indietro e ritrasmette 1 e 2.

6.3 Riscontri cumulativi

Definizione – Riscontro cumulativo

Nel Go-Back-N l'ACK con numero k significa «ho ricevuto correttamente **fino a** $k-1$, aspetto k». Un ACK riscontra quindi anche i frame precedenti, i cui ACK si sono persi.

Con $W = 4$, A invia i frame da 0 a 3; gli ACK 1, 2 e 3 si perdono e arriva solo l'ACK 4. La base passa direttamente a 4, e la finestra diventa 4–7: l'ACK 4 garantisce che anche 0, 1 e 2 sono arrivati. L'Esercizio 4 usa questa regola.

6.4 Frame fuori ordine

L'ESN spiega anche che cosa succede se i frame arrivano in un ordine diverso da quello di invio. Con $W = 3$, il frame 0 è molto in ritardo e i frame 1 e 2 arrivano prima:

1. B scarta 1 e 2, fuori sequenza, e risponde due volte con l'ACK 0, perché il suo ESN è ancora 0;
2. A riceve gli ACK 0 e non fa nulla: la finestra resta 0–2 (l'RTO non è ancora scaduto);
3. arriva il frame 0: B lo accetta e invia l'ACK 1; la finestra di A passa a 1–3 e A può inviare il 3.

I frame 1 e 2, scartati da B, partono di nuovo allo scadere dell'RTO.

6.5 Quale finestra

La finestra migliore è la **minima** che dà $\eta = 1$, cioè $W = W_{\min}$ della Sezione 5. Una finestra più grande non aumenta l'efficienza, che è già 1, ma quando un frame si perde fa ritrasmettere più frame. L'Esercizio 3 calcola la finestra ottima e il tempo di trasferimento.

6.6 Numeri di sequenza a m bit

Il numero di sequenza è un campo dell'intestazione, quindi ha un numero fisso di bit, m. Con m bit si scrivono 2^m numeri, da 0 a $2^m - 1$; dopo $2^m - 1$ si riparte da 0, cioè si conta **modulo** 2^m. TCP, per esempio, usa $m = 32$.

La finestra non può essere grande quanto i numeri disponibili. Con $m = 2$ (numeri da 0 a 3) e $W = 4$ (Figura 9, a sinistra):

1. A invia i frame 0, 1, 2 e 3, che arrivano tutti; B li consegna e il suo ESN passa da 0 a 1, 2, 3 e di nuovo 0;
2. tutti gli ACK si perdono: la finestra di A non scorre, e l'RTO del frame 0 scade;
3. A ritrasmette il frame 0. B aspetta proprio un frame 0 e lo accetta come **nuovo**: un duplicato passa allo strato superiore.

Con $W = 3$ (a destra) A invia solo 0, 1 e 2, e dopo l'ultimo l'ESN di B vale 3: il frame 0 ritrasmesso non è quello atteso, e B lo riconosce come duplicato.

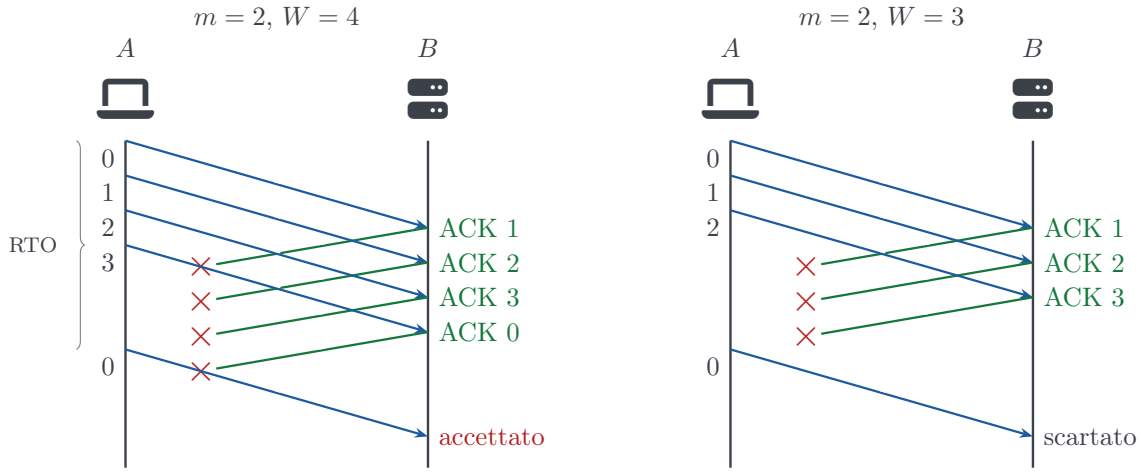


Figura 9: Numeri di sequenza a 2 bit e ACK tutti persi. Con $W = 4$ il frame 0 ritrasmesso arriva quando l'ESN di B è di nuovo 0, e B lo accetta come nuovo. Con $W = 3$ l'ESN è 3, e B lo scarta.

Proposizione – Dimensione massima della finestra

Con numeri di sequenza a m bit, il Go-Back-N funziona se

$$\boxed{W \leq 2^m - 1}.$$

I numeri sono come le ore di un orologio. Se la finestra è grande quanto tutto il quadrante, la lancetta può fare un giro completo e tornare al punto di partenza: B vede di nuovo 0 e non sa se è un frame nuovo o il vecchio. Con un numero in meno la lancetta si ferma prima, e la differenza resta visibile.

La regola si usa anche al contrario. Prima si sceglie la finestra che serve nello scenario, per esempio $W_{\min}$; poi si prende il minimo m con $2^m - 1 \geq W$. Nel collegamento via satellite dell'Esercizio 1, $W = 251$ richiede $2^m \geq 252$, cioè $m = 8$. Un protocollo però fissa m una volta per tutte: TCP usa 32 bit, cioè 4 B in ogni segmento, per coprire tutti gli scenari, da una LAN di pochi metri a un collegamento satellitare.

Il ricevitore del Go-Back-N è anch'esso una finestra, lunga 1: accetta solo l'ESN. Lo stop and wait ha lunghe 1 tutte e due le finestre.

6.7 Piggybacking

Le comunicazioni sono quasi sempre bidirezionali: a una richiesta del client segue la risposta del server. Il **piggybacking** mette l'ACK nell'intestazione di un frame di dati che va nel verso opposto, invece di inviarlo da solo. Ogni frame porta allora due numeri indipendenti:

- il **numero di sequenza**, per il proprio verso;
- l'**ACK number**, che riscontra il verso opposto.

Se la richiesta del client ha numero di sequenza 10, la risposta del server porta ACK 11 e un proprio numero di sequenza, per esempio 5100, che non dipende dal 10.

6.8 Durata del timeout

L'RTO deve lasciare all'ACK il tempo di tornare: la trasmissione del frame, la propagazione in andata, l'elaborazione in B, la trasmissione dell'ACK e la propagazione al ritorno,

$$\text{RTO} \geq T_{\text{TX}} + 2\,T_{\text{P}} + T_{\text{proc}} + T_{\text{ACK}}.$$

- Un RTO **troppo breve** scade prima che l'ACK possa arrivare: i timeout prematuri causano ritrasmissioni inutili.
- Un RTO **troppo lungo** lascia A fermo ad aspettare un ACK che non arriverà.

I tempi reali sono aleatori: le due propagazioni sono uguali su un solo collegamento, ma non lungo un percorso tra due host. Per questo i protocolli stimano il ritardo e adattano l'RTO. Come nella Sezione 5, la finestra deve tenere occupato il canale per tutto l'RTO.

7 Il protocollo Selective Repeat

7.1 Il limite del Go-Back-N

Senza errori un Go-Back-N con $W \geq W_{\min}$ ha $\eta = 1$. Una sola perdita, però, fa ritrasmettere tutta la finestra, anche i frame arrivati corretti, e con un BER alto le ritrasmissioni inutili si moltiplicano. Il **Selective Repeat** (SR) ritrasmette solo i frame persi o sbagliati.

Definizione – Selective Repeat: regole di A e di B

Trasmettitore A: ha una finestra di W_s frame, come nel Go-Back-N, ma un **timer per ogni frame** in attesa di riscontro. Se il timer di un frame scade, A ritrasmette **solo quel frame**. Un NAK fa ritrasmettere il frame più vecchio non riscontrato.

Ricevitore B: ha una finestra di $W_r > 1$ frame, che parte dall'ESN. Un frame corretto fuori sequenza ma **dentro la finestra** non si scarta: si mette in memoria, **ricevuto ma non consegnato**. Quando arriva il frame mancante, B consegna in ordine tutti i frame consecutivi e la finestra scorre. Un frame fuori dalla finestra si scarta, perché non c'è posto dove salvarlo.

Riscontri: l'ACK porta sempre l'ESN, cioè la base della finestra di B. Un frame fuori sequenza fa ripetere lo stesso ACK, che segnala il buco.

Nella Figura 10 $W_s = W_r = 5$ e il frame 2 si perde:

1. i frame 0 e 1 arrivano: B risponde con ACK 1 e ACK 2, e la sua finestra parte da 2;
2. i frame 3, 4, 5 e 6 arrivano corretti ma fuori sequenza: B li mette in memoria e risponde ogni volta con l'ESN 2, la prima volta con un NAK;
3. A ha la finestra 2–6 piena e si ferma. Arriva il NAK 2, e A ritrasmette **solo** il 2, il più vecchio non riscontrato (senza NAK lo farebbe allo scadere del suo timer);
4. il 2 arriva: B consegna i frame da 2 a 6, la base passa a 7, e l'ACK 7 riscontra in un colpo tutta la finestra.

Il ricevitore ripete l'ESN anche se un frame è arrivato: gli ACK ripetuti dicono ad A che qualcosa manca. TCP, nelle varianti Reno e New Reno, ne fa una regola, la **fast retransmission**: dopo tre ACK duplicati A ritrasmette subito quel frame, senza aspettare il timeout.

Con due perdite il meccanismo è lo stesso. Con $W_s = W_r = 4$, A invia i frame da 0 a 5; si perdono il 2 e il 4:

1. 0 e 1 arrivano: ACK 1 e ACK 2;
2. il 3 arriva fuori sequenza e va in memoria: ACK 2; il 5 arriva, va in memoria: ancora ACK 2, perché manca il 2;
3. allo scadere del suo timer il 2 viene ritrasmesso e arriva: B consegna 2 e 3, la base passa a 4, e l'ACK è 4. Il 5 resta in memoria finché non arriva il 4.

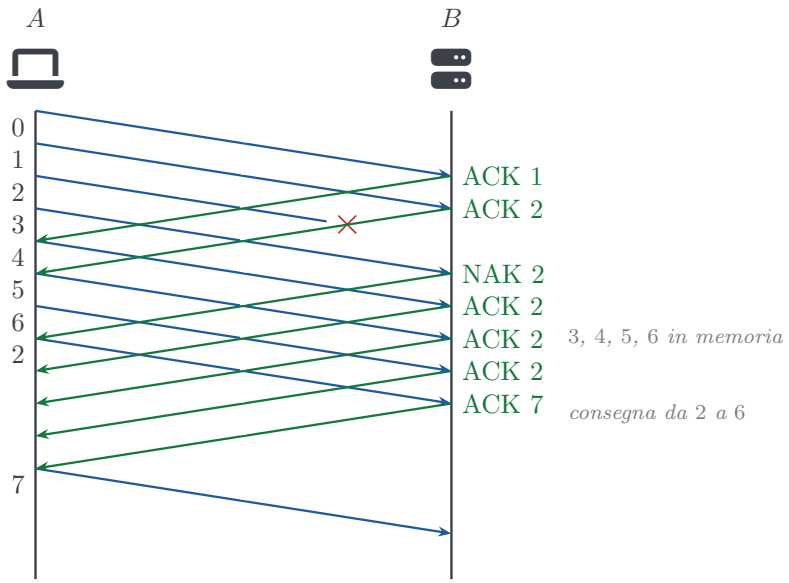


Figura 10: Selective Repeat con $W_s = W_r = 5$ e il frame 2 perso. B tiene in memoria i frame da 3 a 6 e ripete l'ESN 2. Al NAK A ritrasmette solo il 2; quando arriva, B consegna cinque frame e risponde con l'ACK 7.

Proposizione – Dimensione delle finestre nel Selective Repeat

Con numeri di sequenza a m bit, le due finestre devono rispettare

$$\boxed{W_s + W_r \leq 2^m}.$$

Con finestre uguali, $W_s = W_r \leq 2^{m-1}$.

7.2 Confronto dei tre protocolli

Senza errori, il Selective Repeat ha le prestazioni del Go-Back-N. Con errori ritrasmette un frame solo quando quel frame è sbagliato, quindi ogni frame richiede in media $1/(1 - P_f)$ trasmissioni e, se la finestra è almeno $W_{\min}$,

$$\eta_{\text{SR}} = 1 - P_f.$$

Per il Go-Back-N, che dopo un errore ritrasmette la finestra, si dimostra che

$$\eta_{\text{GBN}} = \frac{1 - P_f}{1 + (W - 1)\, P_f}.$$

Esempio – Efficienza e BER

$L = 10\,000$ bit, $L_H = 200$ bit, $C = 1$ Mbit/s, RTT $= 100$ ms. Il BDP è $100\,000$ bit, cioè 10 frame, e $W = W_{\min} = 11$:

BER	0	10^{-6}	10^{-5}	10^{-4}
Stop and wait	9,07 %	8,98 %	8,21 %	3,34 %
Go-Back-N	100 %	90,0 %	46,4 %	5,0 %
Selective Repeat	100 %	99,0 %	90,5 %	36,8 %

Con BER 10^{-5} il Selective Repeat tiene ancora il 90%, mentre lo stop and wait è all'8% e il Go-Back-N al 46%. Con BER 10^{-4} tutti e tre crollano.

Riepilogo

- **RDT**: consegna senza errori, una volta sola e in ordine, su un canale che può sbagliare, perdere, duplicare e disordinare.
- **End-to-end e hop by hop**: il primo è nel trasporto (TCP) e serve sempre a chi lo chiede; il secondo è nello strato 2, solo dove gli errori sono frequenti.
- **Stop and wait**: un frame alla volta; se l'RTO scade si ritrasmette. Il frame porta il suo numero, l'ACK il prossimo atteso; bastano 0 e 1, con quattro stati $(S_{\text{last}}, R_{\text{next}})$.
- **Efficienza**: $\eta = 1/(1+2CT_{\text{P}}/L+L_H/L) \simeq 1/(1+\text{BDP}/L)$, con $\text{BDP} = C \cdot \text{RTT}$.
- **Errori**: $1-P_f = (1-\text{BER})^L$, in media $1/(1-P_f)$ trasmissioni, $\eta = (1-P_f)\,\eta_{\text{SW}}$. Con perdite: $n\,T_{\text{TOT}} + k\,\text{RTO}$, con l'RTO almeno T_{TOT}.
- **Pipeline**: W frame in volo, $\eta = \min(1, W\eta_{\text{SW}})$, $W_{\min} = \lceil 1/\eta_{\text{SW}} \rceil$.
- **Go-Back-N**: finestra da base a base $+\,W-1$; l'ACK, cumulativo, porta l'ESN e sposta la base. B scarta i fuori sequenza; allo scadere dell'RTO del frame più vecchio si ritrasmette la finestra. Conviene $W = W_{\min}$, con $W \leq 2^m - 1$.
- **Selective Repeat**: un timer per frame, ritrasmette solo i frame persi; B tiene in memoria i fuori sequenza nella sua finestra e l'ACK porta la base. $W_s + W_r \leq 2^m$, $\eta = 1 - P_f$.

8 Esercizi

Esercizio 1 – Un collegamento via satellite

Si consideri un canale via satellite della capacità di 1 Mbit/s. Considerando che il tempo di propagazione attraverso un satellite geostazionario richiede 250 ms, si chiede di dimensionare la minima finestra di trasmissione di un protocollo Go-Back-N (con time-out) in modo che sia consentita la massima efficienza temporale del canale quando vengano trasmesse trame di 2000 bit in assenza di errori. Si suppongano gli ACK trascurabili.

Soluzione a pagina 23 →

Esercizio 2 – Stop and wait con errori e riscontri negativi

Si consideri un'interfaccia tra un'entità A e un'entità B in cui si implementa un protocollo di recupero di errore del tipo stop and wait. Si assuma che:

i) siano possibili solo errori nel verso A → B;

ii) la probabilità di rivelare un'unità informativa errata sia $p = 0{,}25$;

iii) la capacità di trasferimento dell'interfaccia sia asimmetrica, in particolare sia $C_1 = 100\,\text{kbit/s}$ nel verso A $\rightarrow$ B e C_2 (incognita) nel verso B $\rightarrow$ A;

iv) il ritardo di propagazione sia di 2 ms;

v) le unità informative siano di dimensione pari a 400 B mentre i riscontri siano di dimensione pari a 30 B;

vi) B emetta sempre riscontri sia positivi che negativi.

Considerando esclusivamente la trasmissione nel verso da A a B, si chiede di:

1. tracciare il diagramma spazio-tempo relativo all'emissione di tre unità informative assumendo che la seconda giunga errata a B (la prima volta in cui viene emessa);
2. calcolare il valore minimo della capacità C_2 del canale di ritorno (B $\rightarrow$ A) tale per cui la portata media normalizzata dell'interfaccia sia maggiore o uguale a 0,5.

Soluzione a pagina 23 →

Esercizio 3 – Appello del 11/02/2025, Canale 1, Prof.ssa Francesca Cuomo, esercizio

Si consideri un collegamento tra due terminali A e B con velocità di trasmissione sul link di 10 kbit/s. Il terminale A deve inoltrare a B 7 trame, tutte della stessa dimensione (1000 B), che impiegano un tempo di trasmissione T. Il ritardo di propagazione sul collegamento è $\tau = 2\,\text{s}$ in ognuna delle due direzioni. Si considerano trascurabili i tempi di trasmissione dei riscontri.

I terminali adottano un meccanismo di controllo d'errore basato su stop and wait. Si assuma che il timeout associato a una trama sia pari al tempo minimo necessario per ricevere il relativo riscontro.

Supponendo che nel trasferimento delle 7 trame da A a B la 3ª trama venga persa, si calcoli:

1. il tempo necessario per trasferire le 7 trame, cioè il tempo dall'invio del primo bit della prima trama al riscontro dell'ultima trama;
2. l'efficienza di utilizzo del collegamento;
3. volendo massimizzare l'efficienza, si può adottare un meccanismo Go-Back-N. Si calcolino la dimensione ottima N della finestra di trasmissione e il tempo per trasferire le 7 trame nel caso in cui non ci siano errori o perdite di trame, e si disegni lo schema della trasmissione delle 7 trame.

Soluzione a pagina 24 →

Esercizio 4 – Appello del 08/02/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 3

In una procedura di recupero d'errore Go-Back-N si adotta una finestra in emissione che ha larghezza 4. Supponendo che dal lato emettitore la finestra abbia in un dato istante limite inferiore uguale a 4, come reagisce A ricevendo dal ricevente (B) una PDU con numero di sequenza in ricezione $R_{\text{next}} = 8$?

a. Aggiorna il limite inferiore della finestra al valore 7 e trasmette le trame 7-8.
b. Posiziona il limite inferiore della finestra a 5 e trasmette le trame 5-8.
c. Aggiorna il limite inferiore della finestra al valore 8 e trasmette le trame 8-11.
d. Trasmette per la prima volta le trame 4-7.

Soluzione a pagina 25 →

Esercizio 5 – Appello del 12/01/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 7

Si consideri un protocollo di controllo d'errore di tipo Go-Back-N. Sia: 1) il ritardo di propagazione pari a 1/3 del tempo di trasmissione di una trama; 2) il tempo di trasmissione dei riscontri pari a 2/3 del tempo di trasmissione di una trama e l'elaborazione nei nodi trascurabile. Quanto deve essere la dimensione minima della finestra in emissione W_s affinché, se non si verificano errori, il trasmettitore sia in grado di trasmettere trame continuamente (senza soluzione di continuità)?

a. $W_s = 3$
b. $W_s = 4$
c. $W_s = 2$

Soluzione a pagina 25 →

Esercizio 6 – Appello del 08/02/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 12

Si consideri un protocollo di controllo d'errore di tipo Go-Back-N. Sia: 1) il ritardo di propagazione pari a 1/5 del tempo di trasmissione di una trama; 2) il tempo di trasmissione dei riscontri pari a 3/5 del tempo di trasmissione di una trama e l'elaborazione nei nodi trascurabile. Quanto deve essere la dimensione minima della finestra in emissione W_s affinché, se non si verificano errori, il trasmettitore sia in grado di trasmettere trame continuamente (senza soluzione di continuità)?

a. $W_s = 3$

b. $W_s = 4$

c. $W_s = 2$

Soluzione a pagina 26 →

9 Soluzioni

Soluzione dell'Esercizio 1 – Un collegamento via satellite

I dati sono $C = 1\,\text{Mbit/s}$, $T_\text{P} = 250\,\text{ms}$ (un satellite geostazionario è a circa $36\,000\,\text{km}$), $L = 2000\,\text{bit}$ e $T_\text{ACK} = 0$: l'ACK si trasmette in un istante, ma si propaga. Allora

$$T_\text{TX} = \frac{L}{C} = 2\,\text{ms}, \qquad T_\text{TOT} = T_\text{TX} + 2\,T_\text{P} = 502\,\text{ms}.$$

L'efficienza è massima, $\eta = 1$, se la finestra copre il ciclo: $W\,T_\text{TX} \geq T_\text{TOT}$. La finestra minima è

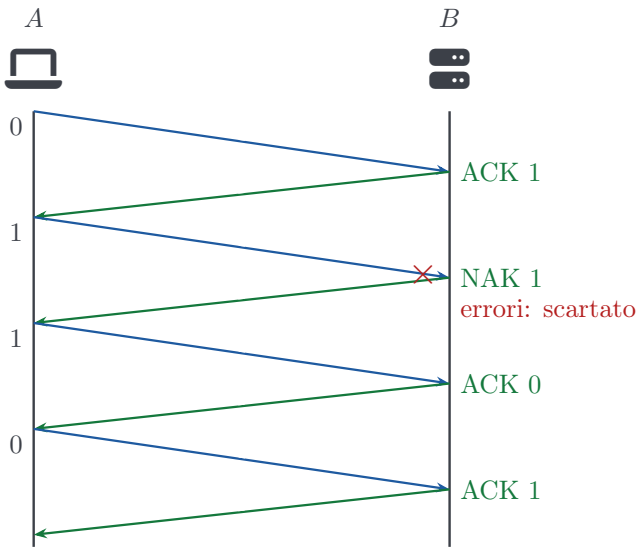
$$W_\text{min} = \left\lceil 1 + \frac{2\,C\,T_\text{P}}{L} \right\rceil = \left\lceil 1 + \frac{2 \cdot 10^6\,\text{bit/s} \cdot 0{,}25\,\text{s}}{2000\,\text{bit}} \right\rceil = \boxed{251}.$$

W è un numero di frame, senza unità di misura; TCP misura invece la finestra in byte. Con $W = 251$ i numeri di sequenza richiedono almeno $m = 8$ bit (Sezione 6.6).

← *Torna all'Esercizio 1, pagina 19*

Soluzione dell'Esercizio 2 – Stop and wait con errori e riscontri negativi

1. Diagramma. I numeri di sequenza sono a un bit. Il frame 0 arriva e B risponde ACK 1. Il frame 1 arriva con errori: B lo scarta e risponde con un NAK, che porta ancora 1, perché l'ESN non cambia. A ritrasmette il 1, che arriva corretto (ACK 0); poi invia la terza unità, con numero 0 (ACK 1).



2. Capacità minima del ritorno. La portata media normalizzata è l'efficienza: tempo

utile diviso tempo totale medio. Con $L = 3200\,\text{bit}$ e $L_H = 240\,\text{bit}$, un ciclo dura

$$T_{\text{TOT}} = \frac{L}{C_1} + \frac{L_H}{C_2} + 2\,T_{\text{P}} = 32\,\text{ms} + \frac{L_H}{C_2} + 4\,\text{ms}.$$

Ogni ciclo si chiude con un ACK o un NAK, senza timeout, e servono in media $1/(1-p)$ cicli (Sezione 4.1, con $P_f = p$). Quindi

$$\eta = \frac{L/C_1}{T_{\text{TOT}}/(1-p)} = \frac{(1-p)\,L/C_1}{T_{\text{TOT}}} \geq 0{,}5 \iff T_{\text{TOT}} \leq \frac{0{,}75 \cdot 32\,\text{ms}}{0{,}5} = 48\,\text{ms}.$$

Allora $L_H/C_2 \leq 48\,\text{ms} - 32\,\text{ms} - 4\,\text{ms} = 12\,\text{ms}$, e

$$C_2 \geq \frac{240\,\text{bit}}{12\,\text{ms}} = \boxed{20\,\text{kbit/s}}.$$

← *Torna all'Esercizio 2, pagina 19*

Soluzione dell'Esercizio 3 – Appello del 11/02/2025, Canale 1, Prof.ssa Francesca Cuomo, esercizio

1. Tempo di trasferimento. Una trama è lunga 8000 bit, quindi

$$T = T_{\text{TX}} = \frac{8000\,\text{bit}}{10\,\text{kbit/s}} = 0{,}8\,\text{s}.$$

Gli ACK sono trascurabili ($T_{\text{ACK}} = 0$), quindi il ciclo è

$$T_{\text{TOT}} = T + 2\,\tau = 0{,}8\,\text{s} + 4\,\text{s} = 4{,}8\,\text{s},$$

e il timeout, il tempo minimo per ricevere l'ACK, vale $\text{RTO} = T_{\text{TOT}}$. Le 7 trame richiedono 7 cicli. La terza trama persa costringe ad aspettare un RTO e a ritrasmetterla, cioè $k = 1$:

$$7\,T_{\text{TOT}} + 1 \cdot \text{RTO} = 8 \cdot 4{,}8\,\text{s} = \boxed{38{,}4\,\text{s}}.$$

2. Efficienza. Il tempo utile è quello di trasmissione delle 7 trame, $7\,T = 5{,}6\,\text{s}$, e il tempo totale è 38,4 s:

$$\eta = \frac{7\,T}{7\,T_{\text{TOT}} + \text{RTO}} = \frac{5{,}6\,\text{s}}{38{,}4\,\text{s}} \simeq \boxed{0{,}146}.$$

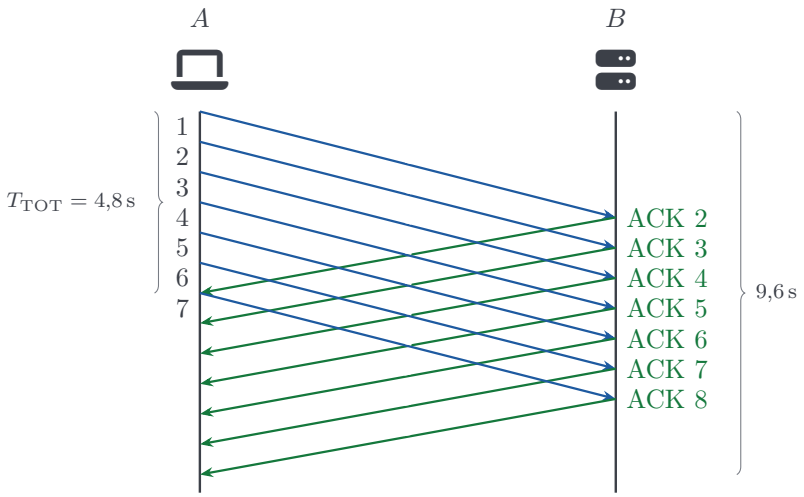
3. Go-Back-N. La finestra ottima N è la minima che dà $\eta = 1$, cioè $W_{\min}$. Il tempo di andata e ritorno della propagazione è $\text{RTT} = 2\tau = 4\,\text{s}$ e $L/C = T = 0{,}8\,\text{s}$, quindi

$$N = W_{\min} = \left\lceil 1 + \frac{C\,\text{RTT}}{L} \right\rceil = \left\lceil 1 + \frac{4\,\text{s}}{0{,}8\,\text{s}} \right\rceil = \lceil 1 + 5 \rceil = \boxed{6}.$$

A invia le trame da 1 a 6 di seguito, in $6T = 4{,}8\,\text{s}$. L'ACK della trama 1 arriva proprio allora, dopo T_{TOT}: la finestra scorre e A invia subito la 7, senza pause.

L'ACK della 7 arriva T_{TOT} dopo l'inizio della sua trasmissione:

$$6\,T + T_{\text{TOT}} = 4{,}8\,\text{s} + 4{,}8\,\text{s} = \boxed{9{,}6\,\text{s}}.$$



Le trame sono numerate da 1 come nel testo; gli ACK portano il numero della prossima trama attesa.

← *Torna all'Esercizio 3, pagina 20*

Soluzione dell'Esercizio 4 – Appello del 08/02/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 3

Il limite inferiore è la base: la finestra è 4–7, e A ha inviato le trame da 4 a 7. L'ACK porta l'ESN 8 ed è cumulativo: B ha ricevuto correttamente fino alla 7. La base passa a 8 e la finestra diventa 8–11: sono tutti numeri liberi, quindi A trasmette le trame da 8 a 11. Risposta: **c**.

La (a) e la (b) mettono la base al numero sbagliato; la (d) ignora l'ACK, e le trame da 4 a 7 erano già state inviate.

← *Torna all'Esercizio 4, pagina 21*

Soluzione dell'Esercizio 5 – Appello del 12/01/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 7

Sia T il tempo di trasmissione di una trama. Allora $T_{\text{P}} = T/3$ e $T_{\text{ACK}} = 2T/3$, e l'elaborazione è trascurabile. Il ciclo è

$$T_{\text{TOT}} = T + 2 \cdot \frac{T}{3} + \frac{2T}{3} = \frac{7}{3}\,T.$$

La trasmissione è continua se la finestra copre il ciclo:

$$W_s \geq \frac{T_{\text{TOT}}}{T} = \frac{7}{3} \simeq 2{,}33, \qquad W_s = \lceil 7/3 \rceil = 3.$$

Con $W_s = 2$ il trasmettitore si fermerebbe, perché $2 < 7/3$; con 4 la finestra sarebbe più grande del necessario. Risposta: a.

← *Torna all'Esercizio 5, pagina 21*

Soluzione dell'Esercizio 6 – Appello del 08/02/2023, Canale 1, Prof.ssa Francesca Cuomo, domanda 12

Con $T_{\text{P}} = T/5$ e $T_{\text{ACK}} = 3T/5$ il ciclo è

$$T_{\text{TOT}} = T + 2 \cdot \frac{T}{5} + \frac{3T}{5} = 2\,T,$$

quindi $W_s \geq T_{\text{TOT}}/T = 2$, e il minimo è $W_s = \lceil 2 \rceil = 2$: due trame occupano $2T$, esattamente il ciclo, e il trasmettitore non si ferma. Risposta: c.

← *Torna all'Esercizio 6, pagina 21*