UNIVERSITA' DEGLI STUDI DI BERGAMO
Dipartimento di Ingegneria

ESERCIZIARIO
"FONDAMENTI DI RETI E TELECOMUNICAZIONE"

Antimo Barbato

Giuseppe Federico Rossi

A.A. 2012/13 - II° Semestre

Indice

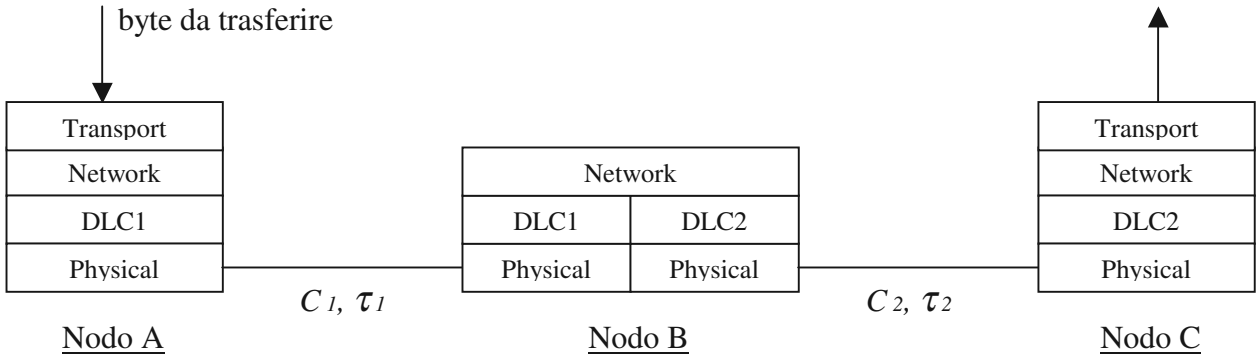
Questo volume è stato realizzato sulla base dell'eserciziario scritto da Luca Anselmi ed Emanuele Goldoni per il corso di Reti di Calcolatori offerto presso l'Università degli Studi di Pavia.

PARTE 1: PARTE GENERALE E SPECIFICA DEL CORSO (6 CFU)

CAPITOLO 1: ANALISI DELLE PRESTAZIONI DELLE RETI DI TLC

Esercizio 1 *(Appello del 27/09/2002)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta i pacchetti in modalità *store-and-forward* con $\tau_{forwarding} = 0$.

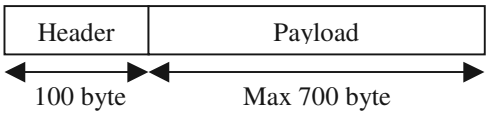


Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):
$C_1 = 16000$ bps $\quad \tau_1 = 75$ ms
C_2 = *da determinare* $\quad \tau_2 = 200$ ms

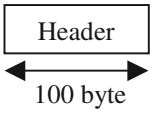
Caratteristiche dei protocolli di comunicazione:

Il livello **Transport** utilizza un protocollo confermato di tipo *Stop-and-Wait*:

PDU-DATI:

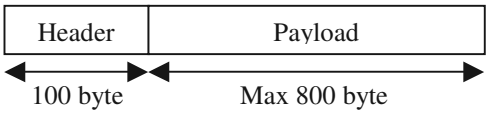


PDU-ACK:



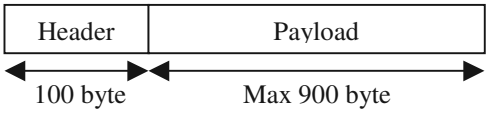
Il livello **Network** utilizza un protocollo non confermato:

PDU-DATI:



I livelli **DLC1** e **DLC2** utilizzano un protocollo non confermato

PDU-DATI:



Domande:
(Disegnare gli schemi temporali di trasferimento dei messaggi giustificando sempre ogni espressione analitica riportata)

1. Calcolare il bit-rate C_2 affinché la capacità del sistema ($C_{SISTEMA}$) sperimentata al di sopra del livello *Transport* sia pari a 280 Byte/s.

2. Utilizzando il valore C_2 calcolato al punto 1, supponendo che la dimensione massima del Payload di DLC1 sia pari a 500 byte anziché 900 byte e sapendo che il protocollo di livello *Network* supporta la frammentazione, calcolare la capacità del sistema ($C_{SISTEMA}$) sperimentata al di sopra del livello *Transport*.

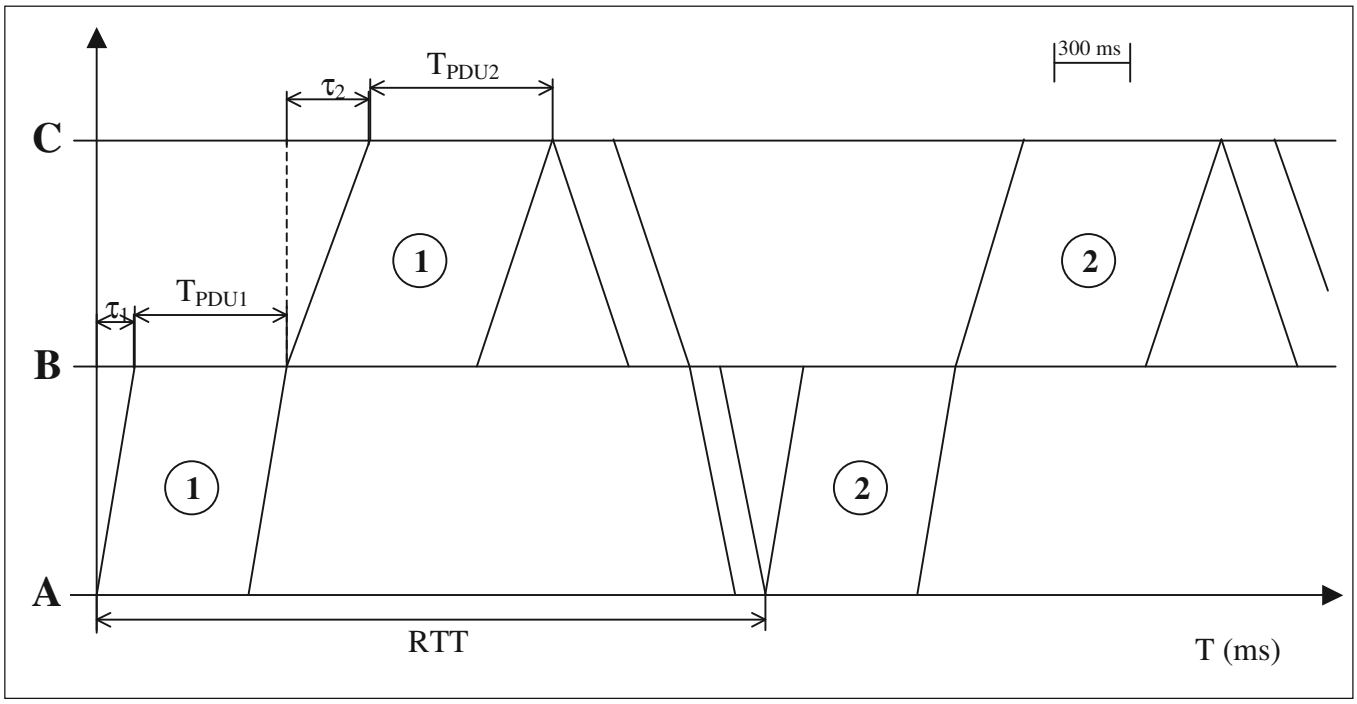
3. (Facoltativa) Quali considerazioni si possono fare sul risultato ottenuto al punto 2, alla luce del fatto che la frammentazione comunque introduce un innalzamento degli overhead di incapsulamento ?

Soluzione

C_1 = 16000 bps = $2000\,Byte/s$ τ_1 = 75 ms = 0,075 s

C_2 = da determinare τ_2 = 200 ms = 0,2 s

1° punto



$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{900+100}{2000} = 0{,}5s$$ Tempo per il trasferimento di una PDU sul canale 1

$$T_{PDU2} = \frac{Payload_{DLC2} + H_{DLC2}}{C_2} = \frac{900+100}{C_2}$$ Tempo per il trasferimento di una PDU sul canale 2

$$T_{ACK1} = \frac{H_{ACK}}{C_1} = \frac{300}{2000} = 0{,}15\,s$$ Tempo per il trasferimento di un'ACK sul canale 1

$$T_{ACK\,2} = \frac{H_{ACK}}{C_2} = \frac{300}{C_2}$$ Tempo per il trasferimento di un'ACK sul canale 2

$$RTT = \tau_1 + T_{PDU\,1} + \tau_2 + T_{PDU\,2} + T_{ACK\,2} + \tau_2 + T_{ACK\,1} + \tau_1 =$$
$$= 0{,}075 + 0{,}5 + 0{,}2 + \frac{1000}{C_2} + \frac{300}{C_2} + 0{,}2 + 0{,}15 + 0{,}075 = 1{,}2 + \frac{1300}{C_2}$$

Sapendo che la capacità del sistema misuratà è pari a 280 Byte/s, è possibile quindi calcolare la capacità del secondo canale:

$$C_{sistema} = \frac{Payload_{TR}}{RTT} = \frac{700}{1{,}2 + \frac{1300}{C_2}}$$

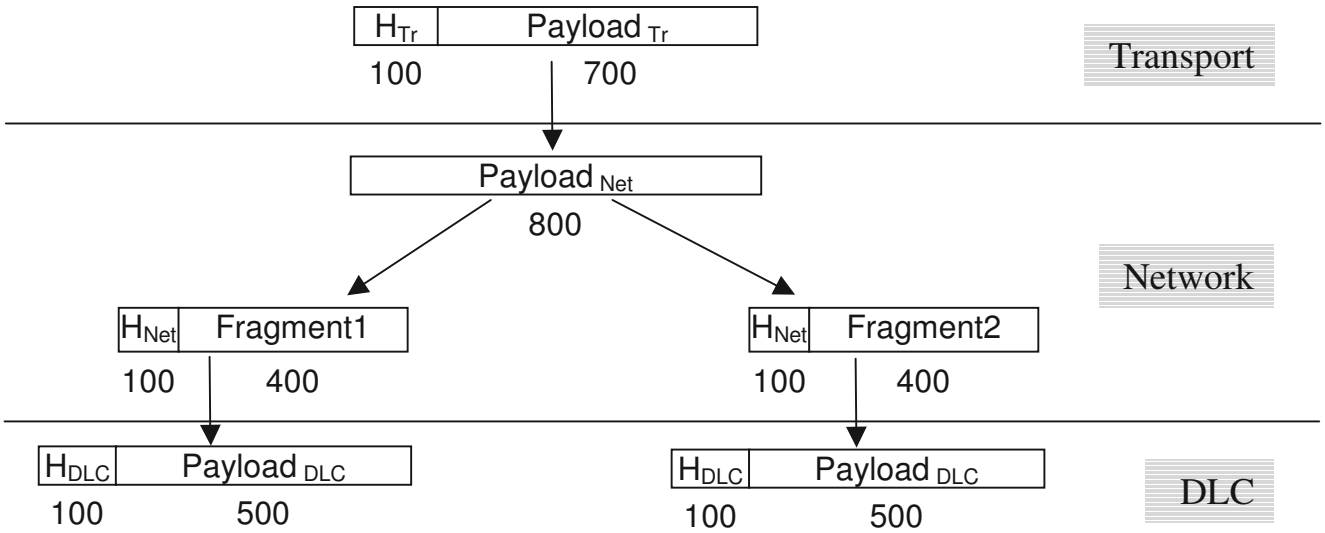
Sostituendo nella formula o riscrevendo il tutto in funzione di C_2 si ottiene infatti:

$$280\,Byte/s = \frac{700}{1{,}2 + \frac{1300}{C_2}}$$

$C_2 = 1000\,Byte/s$ Bit-rate del canale 1

2° punto

Il livello Network deve eseguire la frammentazione nei confronti del DLC1, e quindi anche del DLC2, secondo il seguente schema:



$$T_{Fram1} = \frac{Payload_{fram} + H_{DLC1}}{C_1} = \frac{500+100}{2000} = 0{,}3s$$

Tempo per il trasferimento di un frammento sul canale 1

$$T_{Fram2} = \frac{Payload_{fram} + H_{DLC2}}{C_2} = \frac{500+100}{1000} = 0{,}6s$$

Tempo per il trasferimento di un frammento sul canale 2

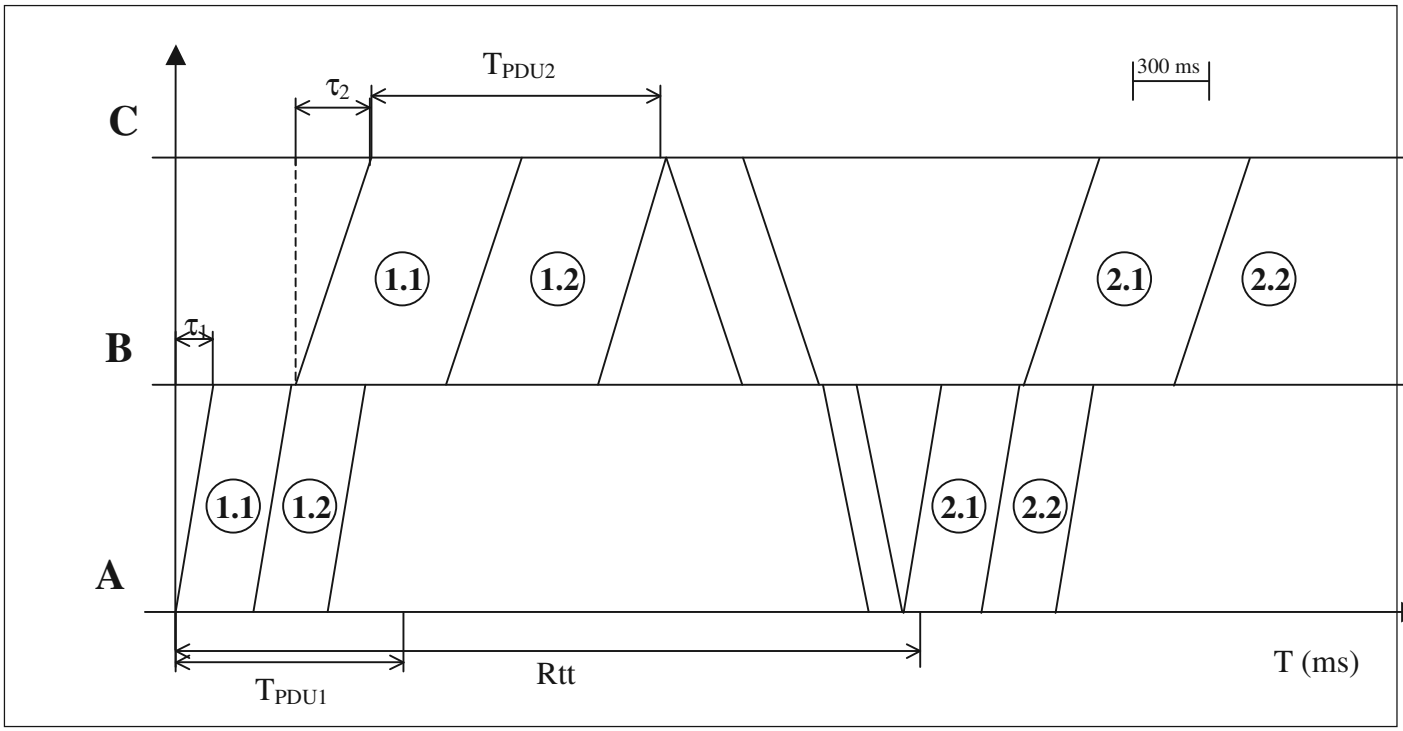
$$T_{ACK1} = \frac{H_{ACK}}{C_1} = \frac{300}{2000} = 0{,}15s$$

Tempo per il trasferimento di un'ACK sul canale 1

$$T_{ACK2} = \frac{H_{ACK}}{C_2} = \frac{300}{1000} = 0{,}3s$$

Tempo per il trasferimento di un'ACK sul canale 2

Anche in questo caso il grafico temporale è fondamentale per trovare il RTT e, di conseguenza, la C del sistema



$$RTT = \tau_1 + T_{Fram1} + \tau_2 + 2 \cdot T_{Fram2} + T_{ACK2} + \tau_2 + T_{ACK1} + \tau_1 =$$
$$= 0{,}075 + 0{,}3 + 0{,}2 + 2 \cdot 0{,}6 + 0{,}3 + 0{,}2 + 0{,}15 + 0{,}075 = 2{,}5s$$

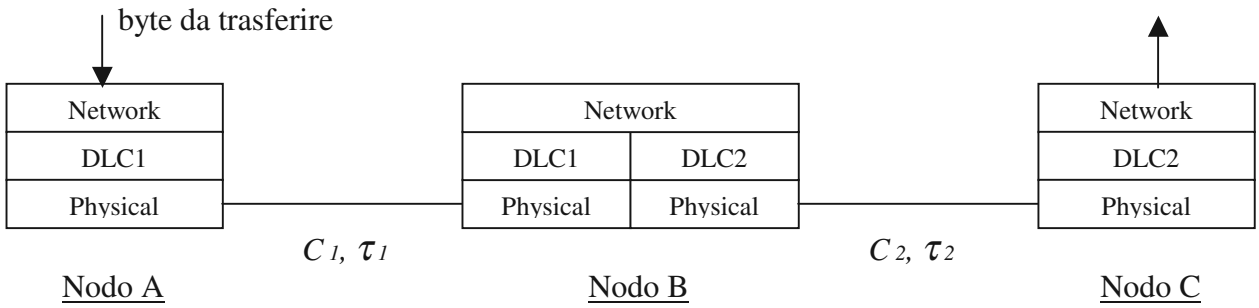
$$C_{sistema} = \frac{Payload_{Tr}}{RTT} = \frac{700}{2{,}5} = 280\,Byte/s$$

3° punto

La frammentazione introduce un overhead di incapsulamento in quanto vengono aggiunti alcuni byte nelle testate dei frammenti. Infatti, per trasferire la stessa payload (700 Byte), utilizzo più byte (200 Byte anziché 100) nel secondo punto rispetto al primo.
Nonostante ciò, si può osservare che la frammentazione non comporta in questo caso alcun aumento/diminuzione della capacità del sistema.
Difatti, dai grafici si nota che il trasferimento del primo pacchetto sul secondo canale nel secondo punto, avviene prima rispetto al trasferimento dell'intera PDU-DATI nel primo esercizio poiché la capacità del secondo canale è inferiore alla capacità del primo.

Esercizio 2 *(1° Itinere del 29/04/2003)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta i pacchetti in un tempo trascurabile con modalità *store-and-forward*. Tutti i nodi dispongono di buffer di dimensione infinita.



Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

$C_1 = 16.000$ bps $\quad \tau_1 = 50$ ms

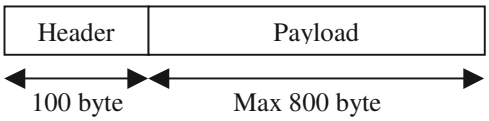
$C_2 = 32.000$ bps $\quad \tau_2 = 100$ ms

Caratteristiche dei protocolli di comunicazione:

Livelli DLC1 e DLC2
Sono specificati più avanti nella sezione **Domande**

Livello Network
Utilizza un protocollo non confermato con possibilità, quando necessario, di frammentazione.

PDU-DATI di Network



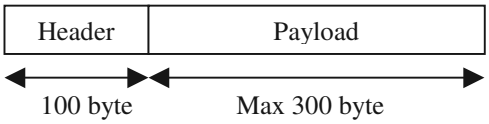
DOMANDE:

Calcolare la capacità del sistema $C_{sistema}$ sperimentata al di sopra del livello *Network* (quando è in corso un trasferimento di byte dal nodo A al nodo C) in ciascuno dei 4 casi indicati qui sotto:

1° Caso:

DLC1 è uguale a DLC2 e viene utilizzato un protocollo non confermato

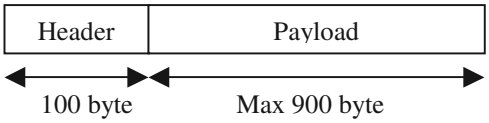
PDU-DATI di DLC1 e DLC2



2° Caso:

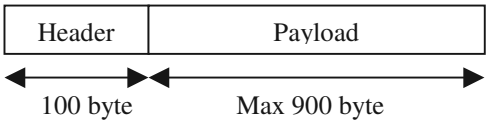
DLC1 utilizza un protocollo non confermato

PDU-DATI di DLC1



DLC2 utilizza un protocollo confermato *Stop-and-Wait*

PDU-DATI di DLC2



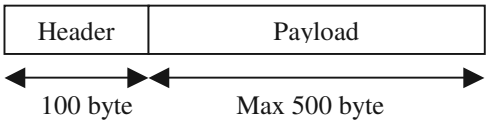
PDU-ACK di DLC2: è costituita dalla sola *header* di 100 byte

3° Caso:

DLC1: come il caso 2

DLC2 utilizza un protocollo confermato *Stop-and-Wait*

PDU-DATI di DLC2



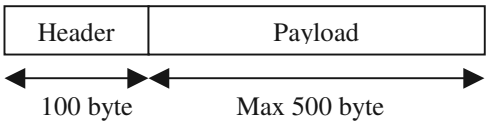
PDU-ACK di DLC2: è costituita dalla sola *header* di 100 byte

4° Caso:

DLC1: come il caso 3

DLC2 utilizza un protocollo confermato *Go-Back-n* con *n=2* (come al solito ipotizzare che l'entità ricevente generi una PDU-ACK per ogni PDU-DATI corretta ricevuta)

PDU-DATI di DLC2



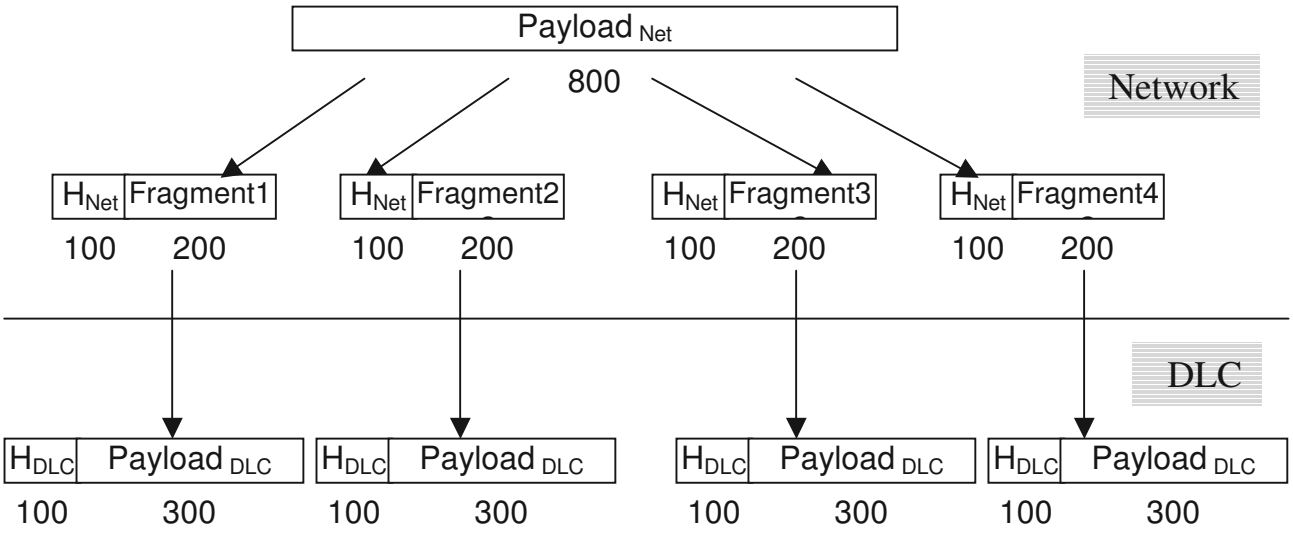
PDU-ACK di DLC2: è costituita dalla sola *header* di 100 byte

Soluzione

C_1 = 16000 bps = 2000 Byte/s τ_1 = 50 ms = 0,05 s
C_2 = 32000 bps = 4000 Byte/s τ_2 = 100 ms = 0,1 s

1° Caso

Il livello Network deve eseguire la frammentazione nei confronti del DLC1 e del DLC2, secondo il seguente schema:



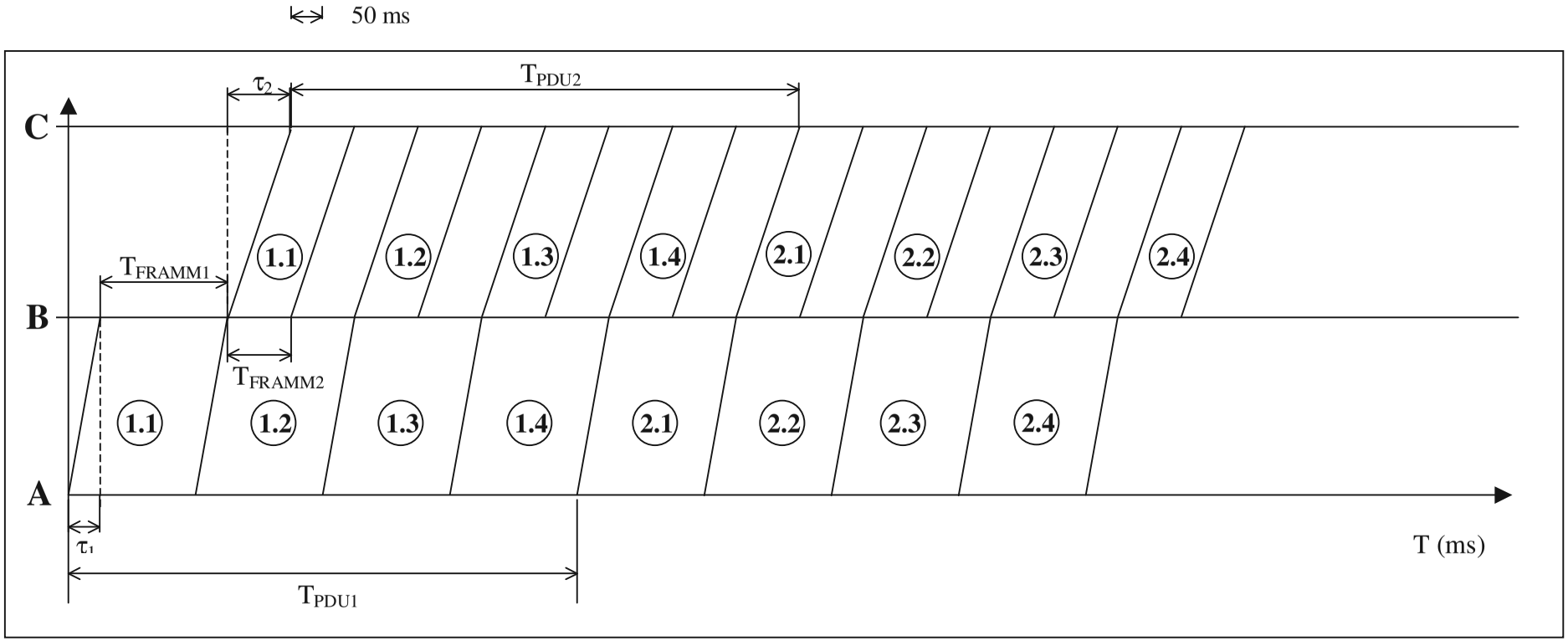
$$T_{FRAMM\,1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{300+100}{2000} = 0{,}2s$$

$$T_{FRAMM\,2} = \frac{Payload_{DLC\,2} + H_{DLC\,2}}{C_2} = \frac{300+100}{4000} = 0{,}1s$$

$$T_{PDU\,1} = T_{FRAMM\,1} \cdot 4 = 0{,}2 \cdot 4 = 0{,}8s$$

Il secondo canale sarebbe più veloce del primo ma si deve adeguare a quest'ultimo attendendo l'arrivo di ogni singolo frammento (vedi schema temporale); pertanto

$$T_{PDU\,2} = T_{PDU\,1}$$



Ricordiamo che la capacità di sistema corrisponde alla capacità relativa al canale strozzante; il canale strozzante è quello con capacità inferiore ossia con T di trasmissione maggiore.
Nel nostro caso, vista la presenza di frammenti, è necessario confrontare il loro tempo di trasmissione: il canale strozzante risulta essere il primo, visto che $T_{framm1} > T_{framm2}$ (vedi schema temporale).
La capacità di sistema verrà quindi calcolata in rapporto al T_{PDU1}.

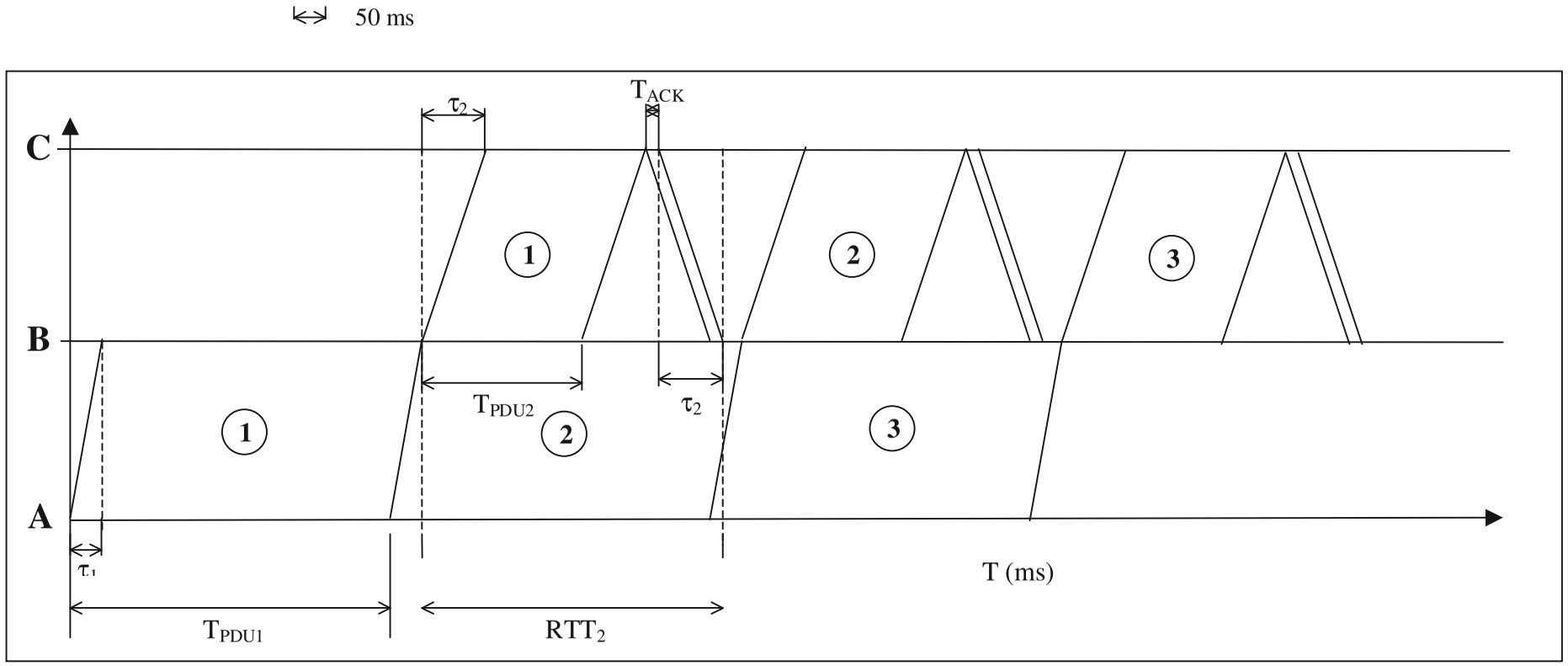
$$C_{sistema} = \frac{Payload_{Net}}{T_{PDU1}} = \frac{800}{0{,}8} = 1000\,Byte/s$$

2° Caso

In questa situazione non è necessario frammentare a livello Network perché il $Payload_{Net}$+ H_{Net} possono essere contenuti nel $Payload_{DLC1}$.

$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{900+100}{2000} = 0{,}5s \qquad T_{ACK} = \frac{Ack}{C_2} = \frac{100}{4000} = 0{,}025s$$

$$T_{PDU2} = \frac{Payload_{DLC2} + H_{DLC2}}{C_2} = \frac{900+100}{4000} = 0{,}25s \qquad RTT_2 = 2\tau_2 + T_{PDU2} + T_{ACK} = 2 \cdot 0{,}1 + 0{,}25 + 0{,}025 = 0{,}475s$$

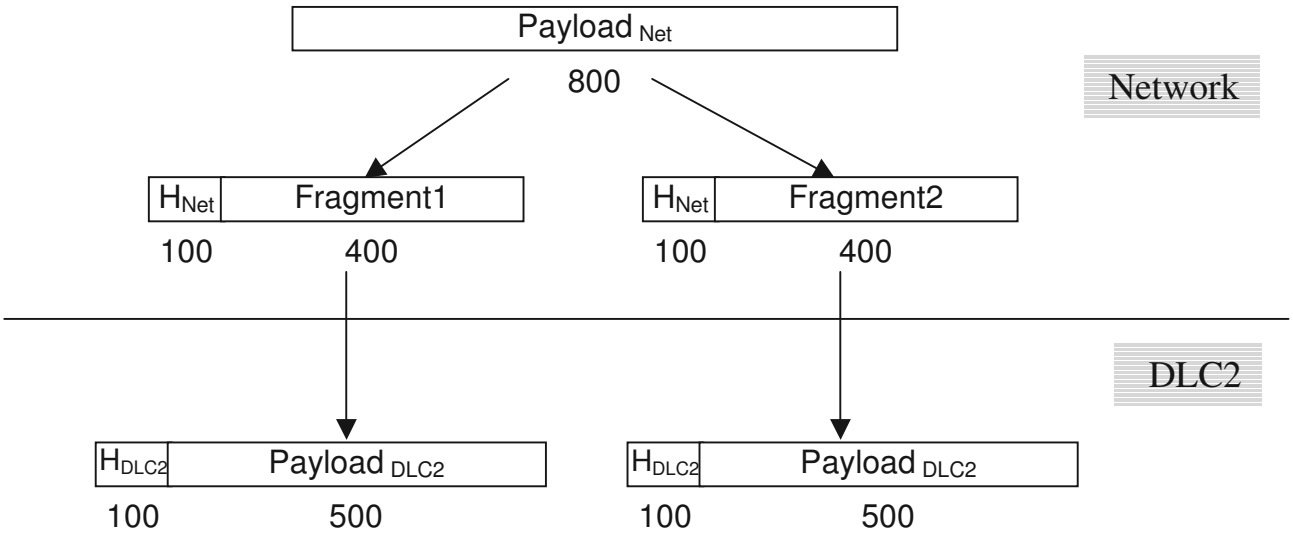


A differenza del 1° Caso, per sapere quale dei due canali è strozzante, occorre fare un confronto tra T_{PDU1} e RTT_2 (non più tra T_{PDU1} e T_{PDU2}).
Nel nostro caso il canale strozzante è quello tra i nodi A e B in quanto $T_{PDU1} > RTT_2$ (vedi schema temporale).

$$C_{sistema} = \frac{Payload_{Net}}{T_{PDU1}} = \frac{800}{0,5} = 1600\, Byte/s$$

3° Caso

In questa situazione non è necessario frammentare a livello Network dal nodo A al nodo B perché il $Payload_{Net} + H_{Net}$ possono essere contenuti nel $Payload_{DLC1}$ mentre è necessario frammentare a livello Network dal nodo B al nodo C.
Il livello Network deve eseguire la frammentazione nei confronti del DLC2, secondo il seguente schema:



$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{900+100}{2000} = 0,5s$$

$$T_{FRAMM2} = \frac{Payload_{DLC2} + H_{DLC2}}{C_2} = \frac{500+100}{4000} = 0,15s$$

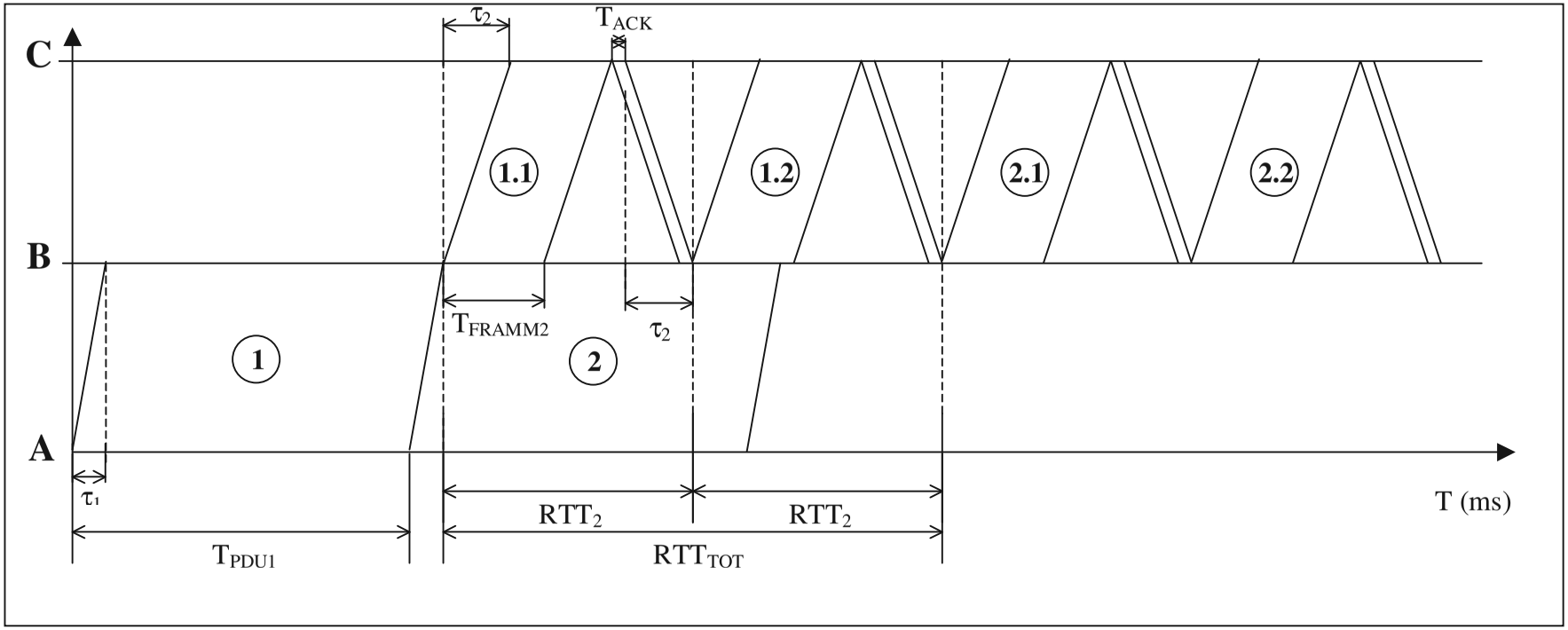
$$T_{ACK} = \frac{Ack}{C_2} = \frac{100}{4000} = 0,025s$$

$$RTT_2 = 2\tau_2 + T_{FRAMM2} + T_{ACK} = 2 \cdot 0,1 + 0,15 + 0,025 = 0,375s$$

$$RTT_{TOT} = 2.RTT_2 = 0,375.2 = 0,75s$$

Per sapere quale dei due canali è strozzante occorre confrontare T_{PDU1} e RTT_{TOT} .
In questo caso il canale strozzante è quello tra i nodi B e C in quanto $T_{PDU1} < RTT_{TOT}$ (vedi schema temporale).

↔ 50 ms



La capacità del sistema dipenderà perciò da RTT_{TOT} riferito al canale B-C.

$$C_{sistema} = \frac{Payload_{Net}}{RTT_{TOT}} = \frac{800}{0,75} = 1066,66\,Byte/s$$

4° Caso

In questo caso è necessaria una frammentazione che è la stessa del 3° caso (vedi grafico precedente).

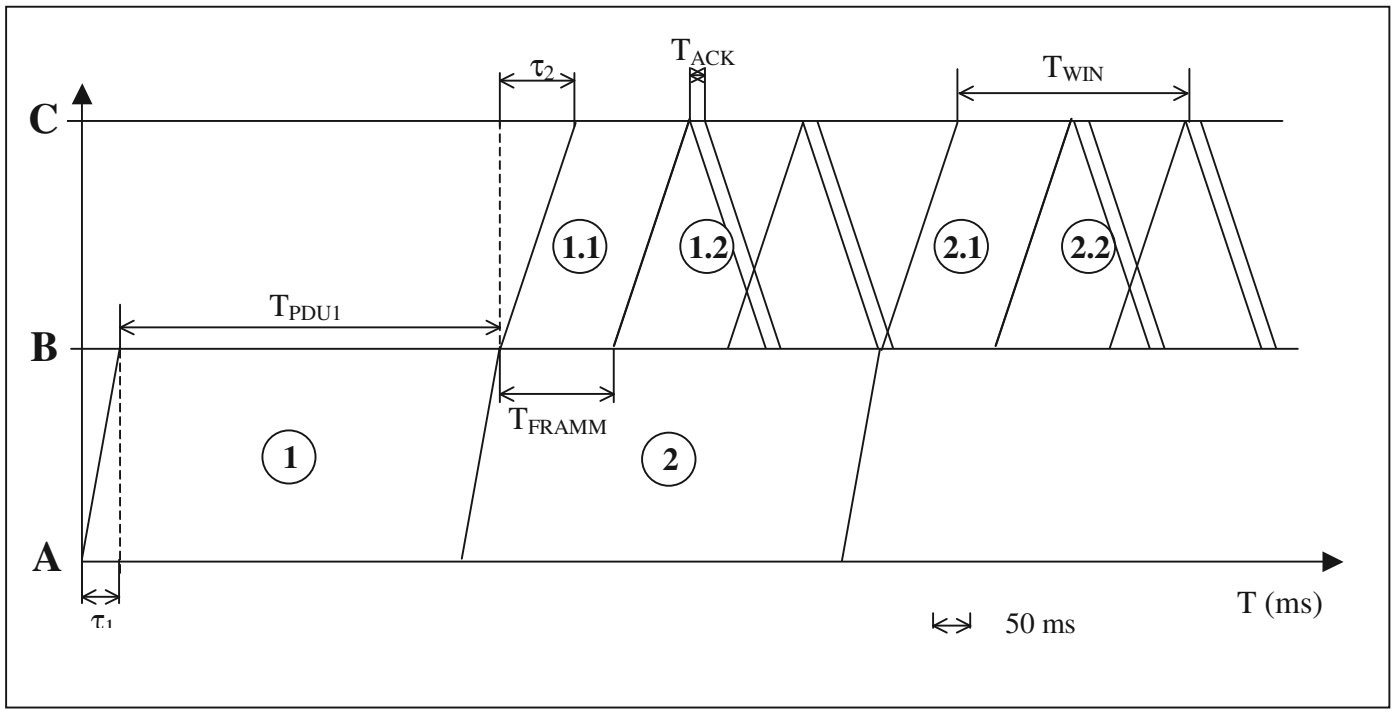
$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{900+100}{2000} = 0{,}5s$$

$$T_{ACK} = \frac{Ack}{C_2} = \frac{100}{4000} = 0{,}025s$$

$$T_{FRAMM2} = \frac{Payload_{DLC2} + H_{DLC2}}{C_2} = \frac{500+100}{4000} = 0{,}15s$$

$$RTT_2 = 2\tau_2 + T_{FRAMM2} + T_{ACK} = 2 \cdot 0{,}1 + 0{,}15 + 0{,}025 = 0{,}375s$$

$$T_{WIN} = 2 \cdot T_{FRAMM2} = 2 \cdot 0{,}15 = 0{,}3s$$

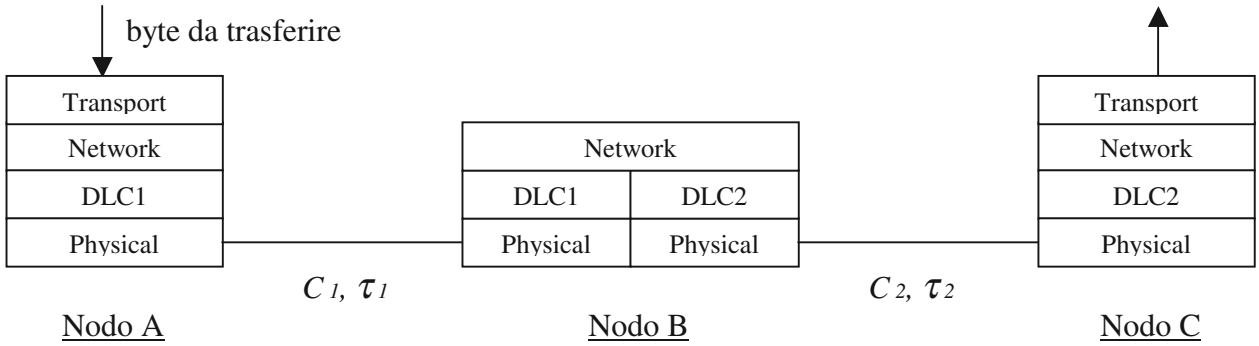


Come si vede dal grafico temporale, i frammenti della seconda PDU non possono partire immediatamente una volta ricevuto l'ACK ma devono attendere che sia stata ricevuta l'intera PDU. Il canale strozzante è quindi il primo, e si ha che:

$$C_{sistema} = \frac{Payload_{Net}}{T_{PDU1}} = \frac{800}{0{,}5} = 1600\,Byte/s$$

Esercizio 3 *(1° Itinere del 29/04/2003)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta i pacchetti in un tempo trascurabile con modalità *store-and-forward*. Tutti i nodi dispongono di buffer di dimensione infinita.



Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

$C_1 = 64.000$ bps $\quad \tau_1 = 50$ ms

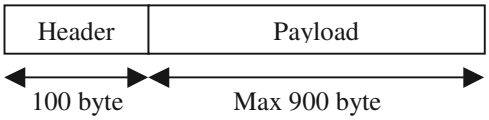
$C_2 = 16.000$ bps $\quad \tau_2 = 100$ ms

Caratteristiche dei protocolli di comunicazione:

Livello DLC1

Utilizza un protocollo non confermato:

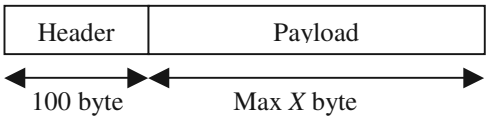
PDU-DATI di DLC1



Livello DLC2

Utilizza un protocollo non confermato:

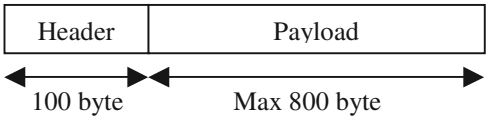
PDU-DATI di DLC2



Livello Network

Utilizza un protocollo non confermato con possibilità, quando necessario, di frammentazione.

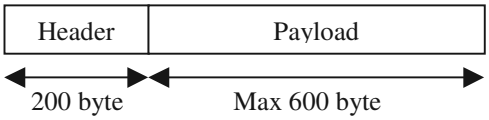
PDU-DATI di Network



Livello Transport

Utilizza un protocollo non confermato:

PDU-DATI di Network



DOMANDA:

Determinare la dimensione massima *X* del *Payload* di DLC2 affinchè la capacità del sistema sperimentata al di sopra del livello *Transport* (quando è in corso un trasferimento di byte dal nodo A al nodo C) sia $C_{sistema}$=*750 byte/s*.

Soluzione

C_1 = 64000 bps = 8000 Byte/s τ_1 = 50 ms = 0,05 s
C_2 = 16000 bps = 2000 Byte/s τ_2 = 100 ms = 0,1 s
$C_{sistema}$=750 Byte/s (imposto)

$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{1000}{8000} = 0{,}125s$$

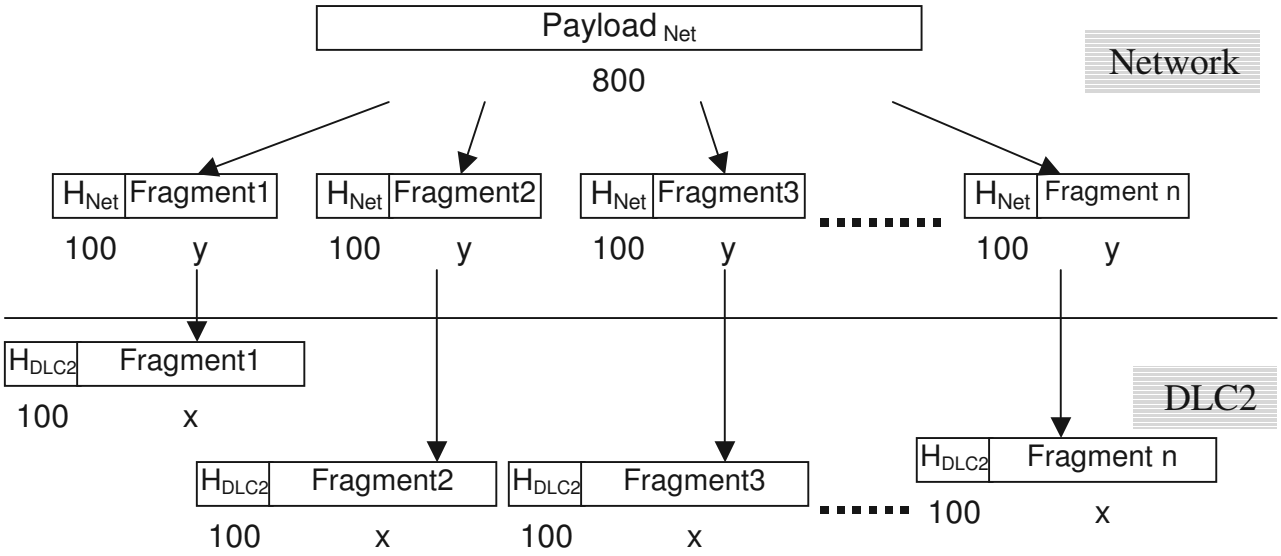
$$C_{sistema} = \frac{Payload_{Tr}}{T_{strozzante}} \longrightarrow \frac{600}{T_{strozzante}} = 750 \longrightarrow T_{strozzante} = \frac{600}{750} \longrightarrow T_{strozzante} = 0{,}8s$$

Il tempo di trasferimento di una PDU sull'ipotetico canale strozzante, in modo che $C_{sistema}$=750 Byte/s, deve essere di 0,8s.

Se α = Byte da trasferire dal nodo B al nodo C in 0,8s:

$$\frac{\alpha}{C_2} = 0{,}8 \;\; \frac{\alpha}{2000} = 0{,}8 \;\; \alpha = 1600Byte$$

Per soddisfare il valore del $T_{strozzante}$ trovato prima, la quantità di dati da trasferire è di 1600 Byte. Ipotizziamo di avere da parte del DLC2 una disponibilità di 1500 Byte del Payload + 100 Byte di Header. Il $Payload_{DLC2}$ non verrà mai riempito completamente, dato che dal Network arrivano solo 900 Byte. Si arriverebbe ad avere a livello DLC2 un pacchetto di 1000 Byte complessivi (H_{NET} + $Payload_{NET}$ + H_{DLC2}), ma non di 1600 Byte.
Si rende necessario quindi il processo di frammentazione: il livello Network frammenta secondo il seguente schema:
n = numero frammenti
y = dimensione Payload del frammento a livello Network
x = dimensione Payload del frammento a livello DLC2



Bisogna fare in modo che la somma di frammenti a livello DLC2 e delle rispettive Header sia di 1600 Byte.

Ora troviamo n:

$$\left[\left(\frac{Payload_{NET}}{n}+H_{NET}\right)+H_{DLC2}\right]=\frac{\alpha}{n}$$

$$\Longleftarrow$$

$$(x+100)=\frac{\alpha}{n}$$

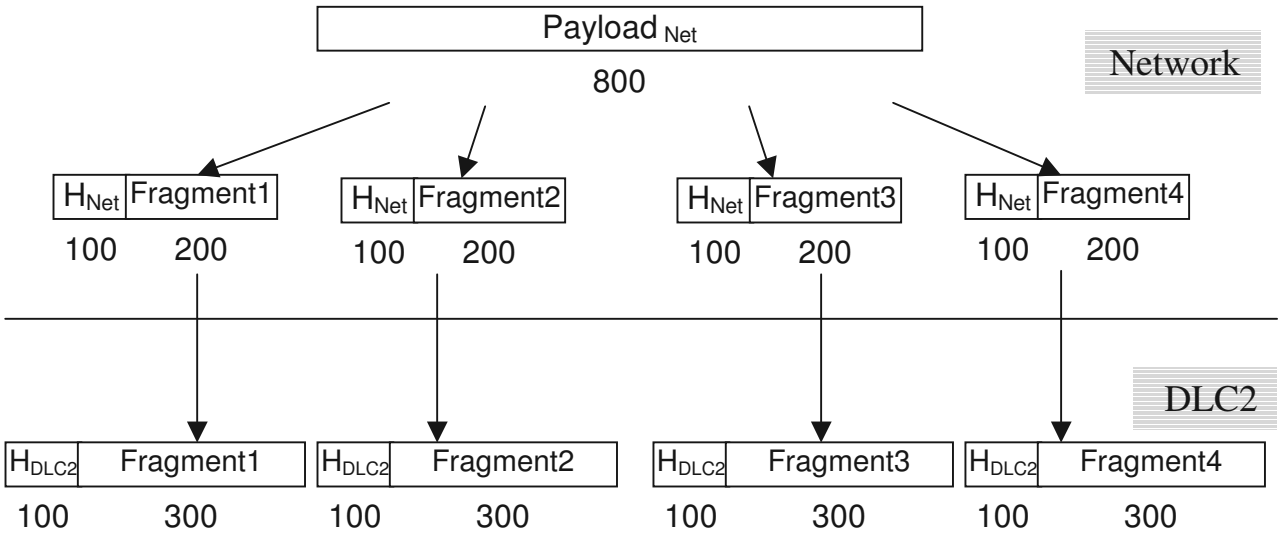
$$[(y+100)+100]=\frac{\alpha}{n}$$

$$\left[\left(\frac{800}{n}+100\right)+100\right]=\frac{1600}{n}$$

$$\left[\frac{800+200\cdot n}{n}\right]=\frac{1600}{n}$$

$$200\cdot n=800 \longrightarrow n=4$$

I frammenti dovranno essere 4.



A questo punto è semplice calcolare le dimensioni di ogni frammento:

$$y = \frac{800}{n} = \frac{800}{4} = 200Byte$$

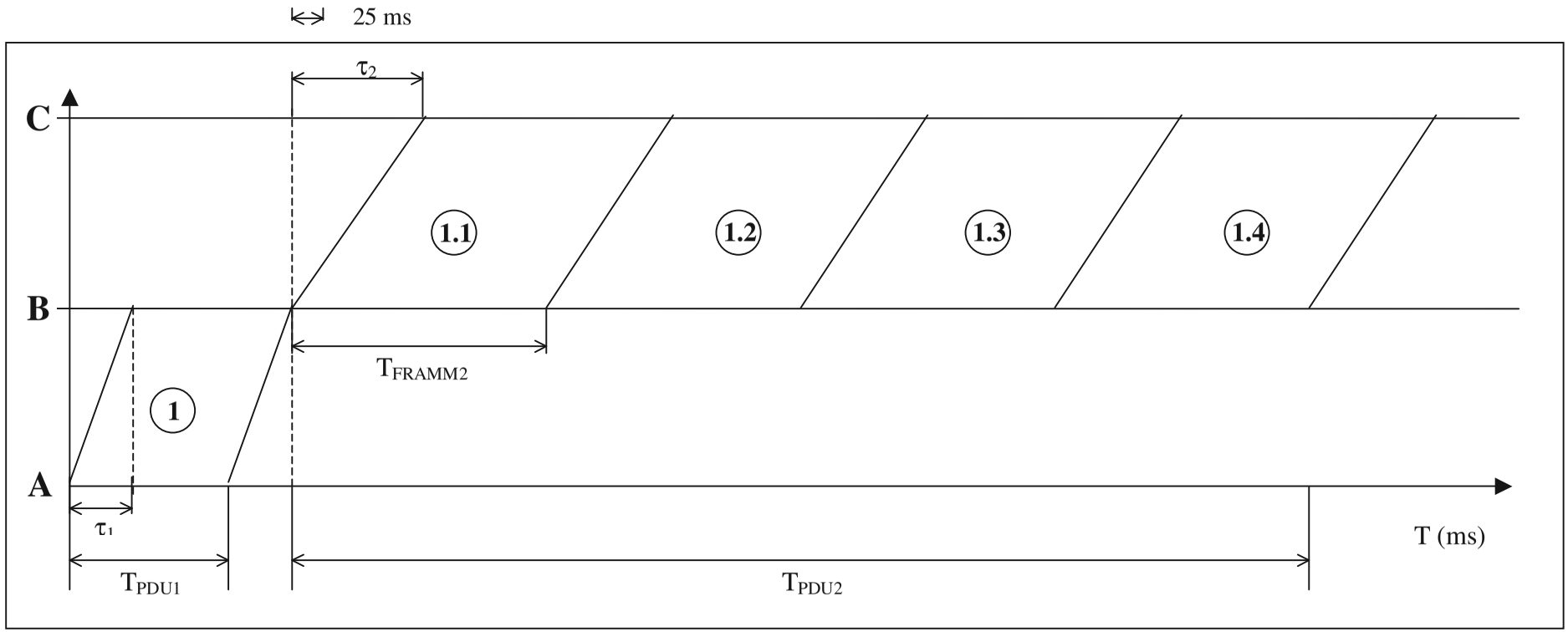
$$x = y + 100 = 200 + 100 = 300Byte$$

Il $Payload_{DLC2}$ avrà quindi una dimensione di 300 Byte.

Calcoliamo ora i tempi di trasferimento sul canale B-C:

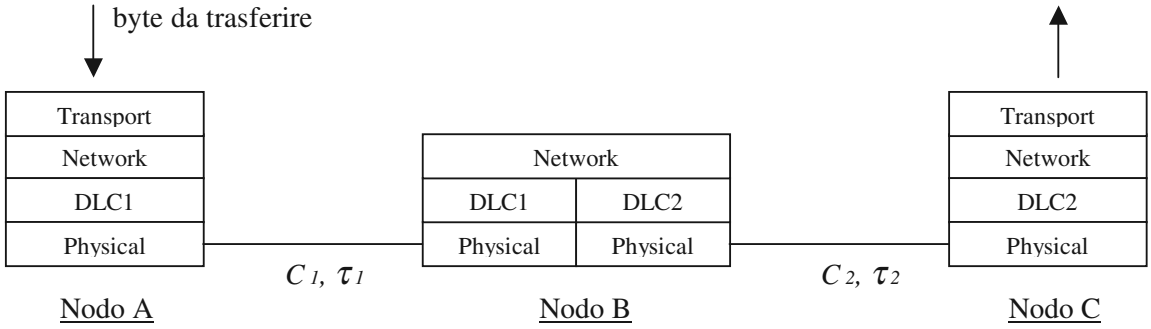
$$T_{FRAMM\,2} = \frac{Payload_{DLC\,2} + H_{DLC\,2}}{C_2} = \frac{300 + 100}{2000} = 0{,}2s$$

$$T_{PDU\,2} = T_{FRAMM\,2} \cdot 4 = 0{,}2 \cdot 4 = 0{,}8s$$ (come stabilito inizialmente)



Esercizio 4 *(Appello del 08/09/2003)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta le PDU di livello 3 in modalità *store-and-forward* con un tempo di elaborazione trascurabile.



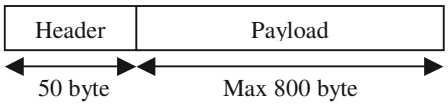
Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

$C_1 = 64.000$ bps $\quad \tau_1 = 50$ ms
$C_2 = 48.000$ bps $\quad \tau_2 = 200$ ms

Caratteristiche dei protocolli di comunicazione:

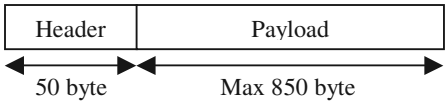
Il livello **Transport** utilizza un protocollo non confermato:

PDU-DATI di Transport



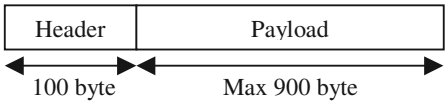
Il livello **Network** utilizza un protocollo non confermato:

PDU-DATI di Network



Il livello **DLC1** utilizza un protocollo confermato *Stop-and-Wait*:

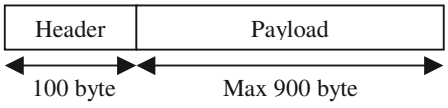
PDU-DATI di DLC1



PDU-ACK di DLC1: è costituita dalla sola *header* di 100 byte

Il livello **DLC2** utilizza un protocollo confermato *Go-Back-n* (con *n* specificato più avanti):

PDU-DATI di DLC2



PDU-ACK di DLC2: è costituita dalla sola *header* di 100 byte

Domande:
(Disegnare sempre tutti gli schemi temporali di trasferimento dei messaggi)

1. Calcolare la capacità del sistema ($C_{sistema}$) sperimentata al di sopra del livello *Transport* al variare del numero intero n. Disegnare inoltre il grafico di $C_{sistema}(n)$.

2. Considerando il caso $n=1$, dire come varia $C_{sistema}$ (aumenta/diminuisce/invariata) nei casi in cui:

   a. C_1 *aumenta (di una quantità infinitesima)*

   b. τ_1 *aumenta (di una quantità infinitesima)*

   c. C_2 *aumenta (di una quantità infinitesima)*

   d. τ_2 *aumenta (di una quantità infinitesima)*

Soluzione

C_1 = 64000 bps = $8000\,Byte/s$ τ_1 = 50 ms = 0,05 s

C_2 = 48000 bps = $6000\,Byte/s$ τ_2 = 200 ms = 0,2 s

$$T_{PDU_1} = \frac{Payload_{DLC_1} + H_{DLC1}}{C_1} = \frac{900+100}{8000} = 0{,}125s$$

$$T_{ACK_1} = \frac{H_{DLC1}}{C_1} = \frac{100}{8000} = 0{,}0125s$$

$$RTT_1 = 2\tau_1 + \frac{Payload_{DLC_1} + H_{DLC1}}{C_1} + \frac{H_{DLC1}}{C_1} = 2 \cdot 0{,}05 + \frac{900+100}{8000} + \frac{100}{8000} = 0{,}2375s$$

$$T_{PDU_2} = \frac{Payload_{DLC_2} + H_{DLC_2}}{C_2} = \frac{900+100}{6000} = 0{,}166s$$

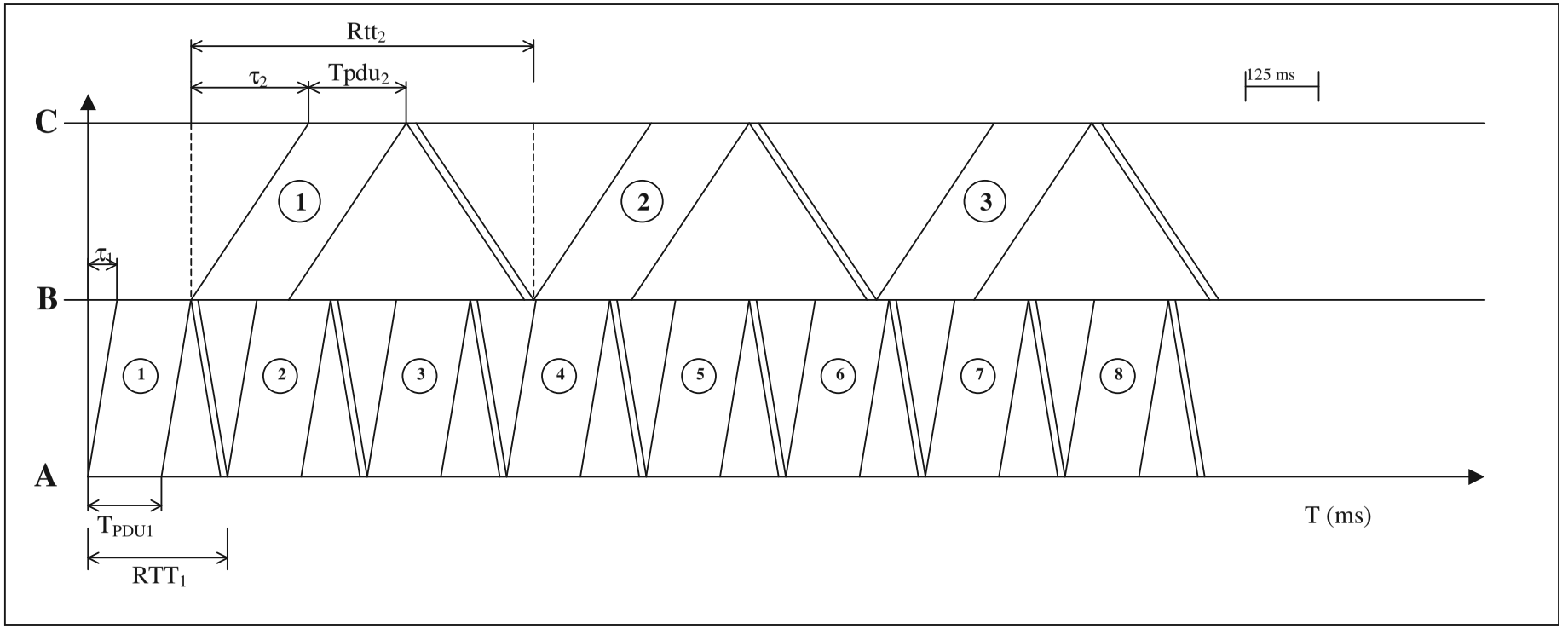
$$T_{ACK_2} = \frac{H_{DLC2}}{C_2} = \frac{100}{6000} = 0{,}0166s$$

$$RTT_2 = 2\tau_2 + \frac{Payload_{DLC_2} + H_{DLC_2}}{C_2} + \frac{H_{DLC2}}{C_2} = 2 \cdot 0{,}2 + \frac{900+100}{6000} + \frac{100}{6000} = 0{,}582s$$

Caso n=1:

Con n=1 il secondo canale Go Back n si comporta come uno Stop & Wait. Siccome $RTT_2 > RTT_1$ è il canale 2 che condiziona il ritmo di arrivo delle PDU;

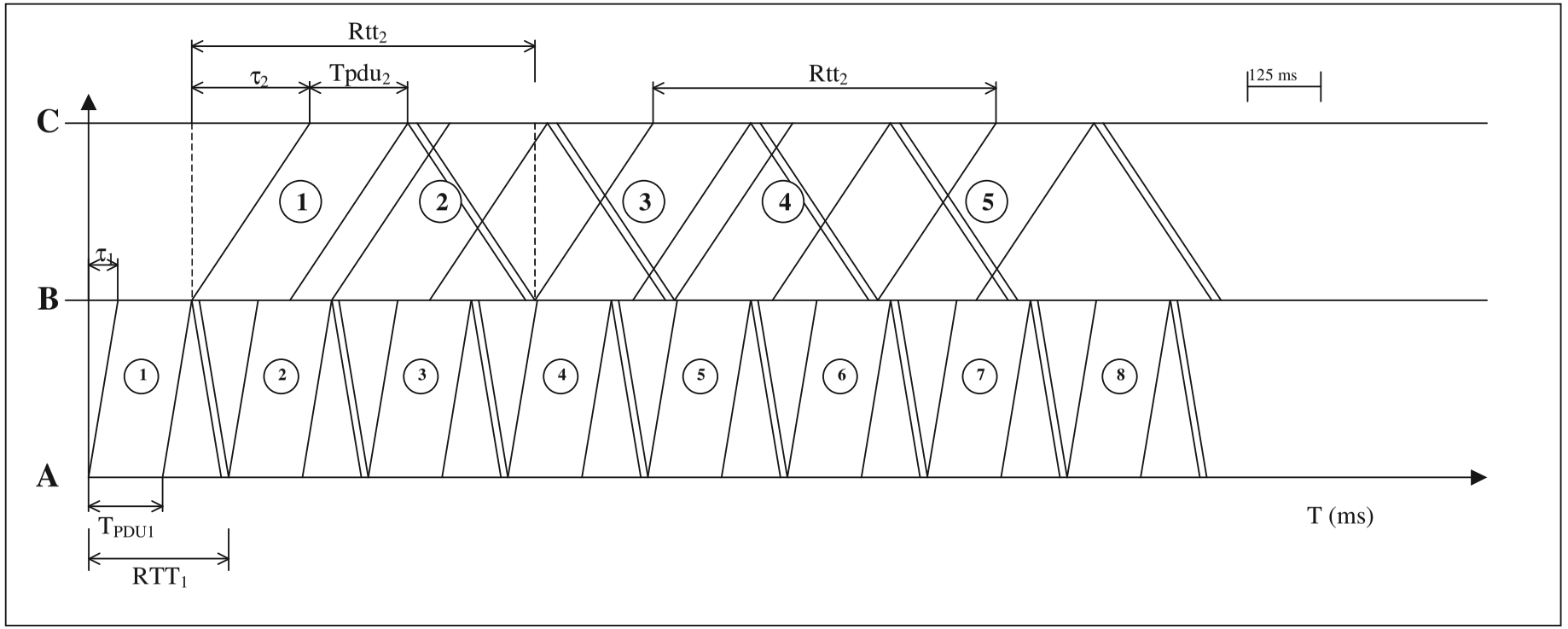
$$C_{sistema}(1) = \frac{Payload_{Tr}}{RTT_2} = \frac{800}{0{,}582} = 1374.5\,Byte/s$$



Caso n=2:

In questo caso sul lato destinatario si ricevono 2 PDU ogni RTT_2;

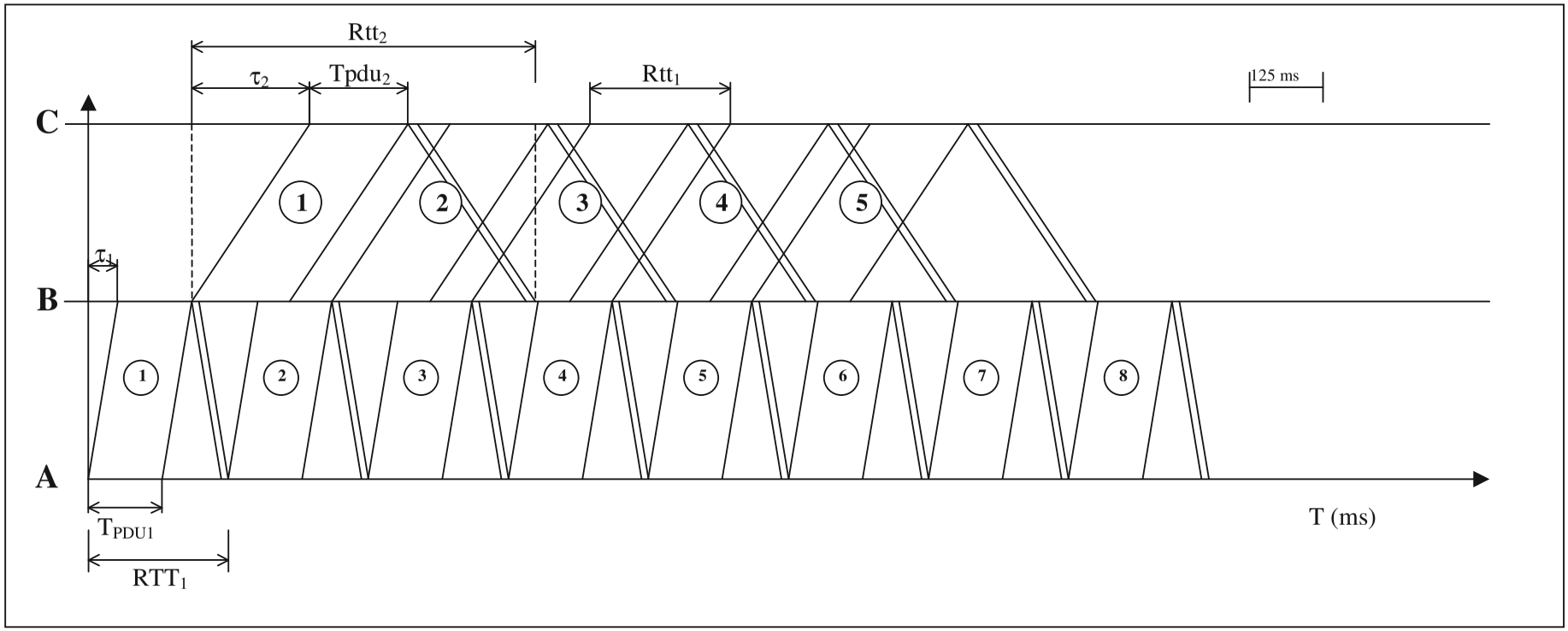
$$C_{sistema}(2) = \frac{2Payload_{Tr}}{RTT_2} = \frac{2 \cdot 800}{0{,}582} = 2749.1 Byte/s$$

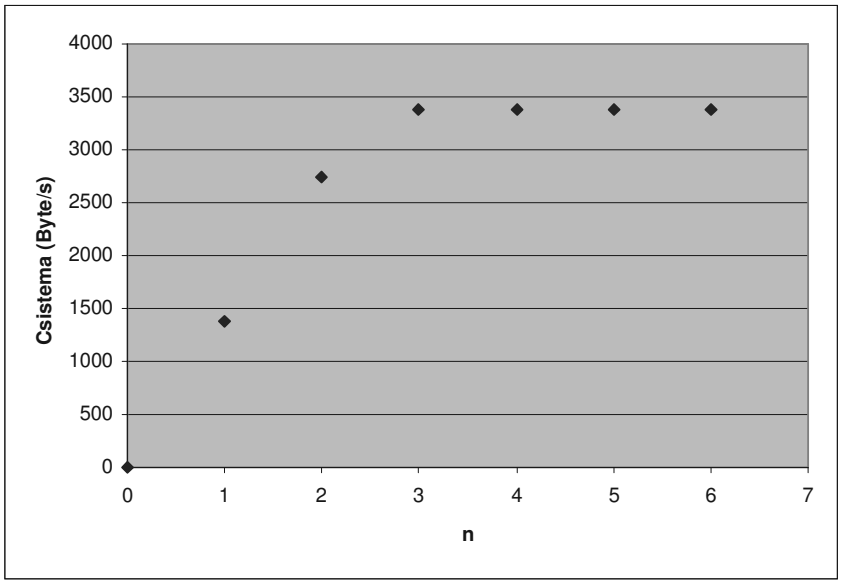


Caso n≥3:

Come si vede dallo schema temporale, la capacità del sistema non dipende più da n, ma dalla velocità del canale 1

$$C_{sistema}(\geq 3) = \frac{Payload_{Tr}}{RTT_1} = \frac{800}{0{,}237} = 3375.5\,Byte/s$$





La funzione $C_{sistema}(n)$

Punto 2:

Nel caso in cui n=1 (caso assimilabile al Go-Back n) la formula trovata è ancora valida se le capacità o ritardi dei canali non subiscono sostanziali variazioni (come accade in questo caso, dove gli spostamenti dal valore iniziale sono infinitesimali). Recuperiamo quindi la formula ricavata in precedenza per il calcolo della C di sistema:

$$C_{sistema}(1) = \frac{Payload_{Tr}}{RTT_2} = \frac{800Byte}{RTT_2}$$

Come si può vedere, la capacità del sistema trovata è funzione di RTT_2, che a sua volta avevamo calcolato in funzione di τ_2 e C_2.

$$RTT_2 = 2\tau_2 + \frac{Payload_{DLC_2} + H_{DLC_2}}{C_2} + \frac{H_{DLC2}}{C_2} = 2 \cdot \tau_2 + \frac{1100}{C_2}$$

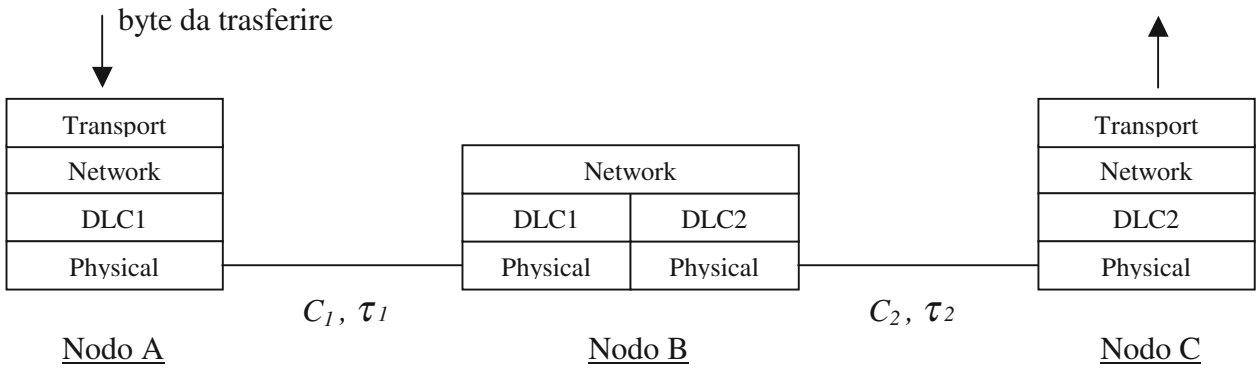
Riscrivendo quindi $C_{sistema}$:

$$C_{sistema} = \frac{Payload_{Tr}}{2 \cdot \tau_2 + \frac{1100}{C_2}}$$

La Capacità del sistema è quindi indipendente da C_1 e τ_1 e pertanto aumenti infinitesimali di questi due valori non avrebbero conseguenze sulla Capacità finale del sistema. Un aumento del ritardo τ del canale 2 diminuirebbe invece la velocità di trasferimento del sistema, mentre un incremento di C_2 ne migliorerebbe le prestazioni.

Esercizio 5 *(1° Itinere del 24/11/2003)*

Sia data la rete indicata in figura (il sistema è privo di errori), nella quale il nodo B commuta i pacchetti a livello 3 in un tempo trascurabile con modalità *store-and-forward*. Tutti i nodi dispongono di buffer di dimensione infinita.



Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

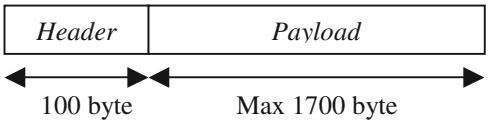
$C_1 = 24.000$ *bps* $\quad \tau_1 = 150$ *ms*

$C_2 = 32.000$ *bps* $\quad \tau_2 = 100$ *ms*

Caratteristiche dei protocolli di comunicazione:

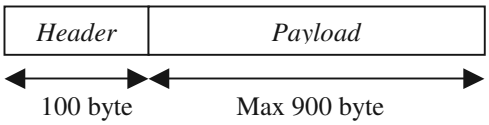
DLC1 utilizza un protocollo non confermato

PDU-DATI di DLC1



DLC2 utilizza un protocollo confermato *Go-Back-n* (come al solito ipotizzare che l'entità ricevente generi una PDU-ACK per ogni PDU-DATI corretta ricevuta)

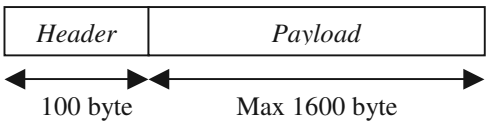
PDU-DATI di DLC2



PDU-ACK di DLC2: è costituita dalla sola *header* di 100 byte

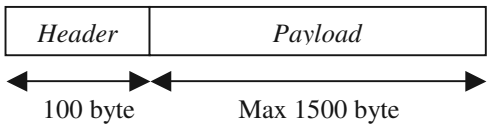
Network utilizza un protocollo non confermato, il quale prevede, quando serve, la funzione di frammentazione.

PDU-DATI di Network



Transport utilizza un protocollo non confermato

PDU-DATI di Transport



Domande:
(Disegnare tutti gli schemi temporali; spiegare sempre ed in dettaglio ogni passo/assunzione fatti)

1. Calcolare la capacità del sistema $C_{sistema}$ sperimentata al di sopra del livello *Transport* quando è in corso un trasferimento di byte dal nodo A al nodo C, nel caso in cui $n=1$ (DLC2)

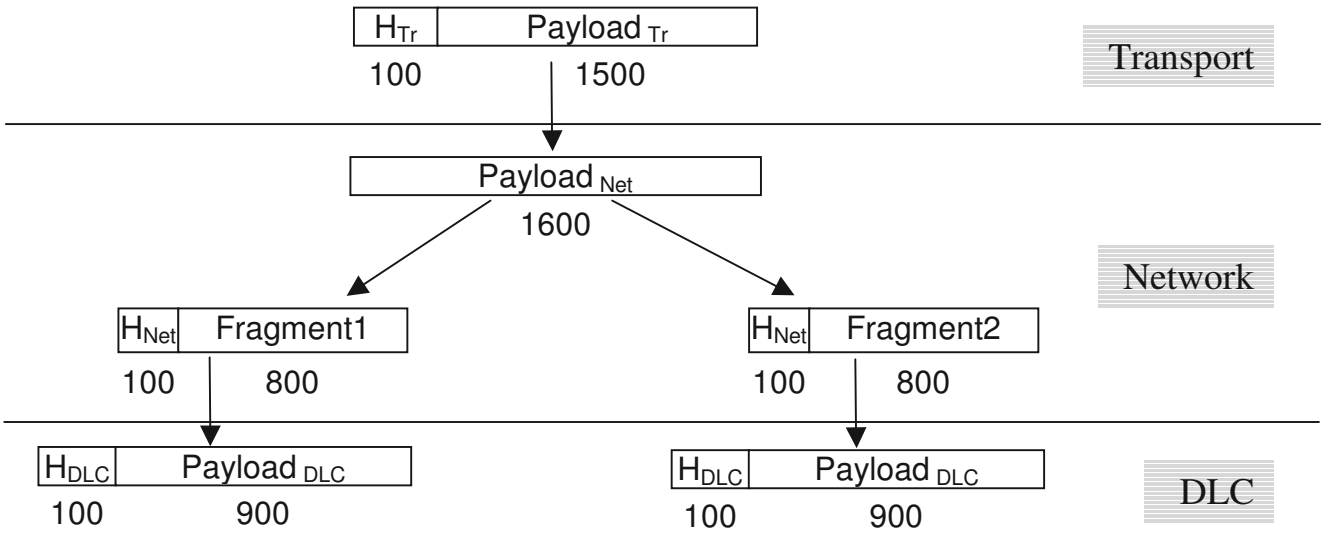
2. Mantenendo ferme le ipotesi di cui al punto 1 e supponendo di far variare C_2 da zero a infinito, calcolare la funzione $C_{sistema}(C_2)$ tracciandone altresì il grafico.

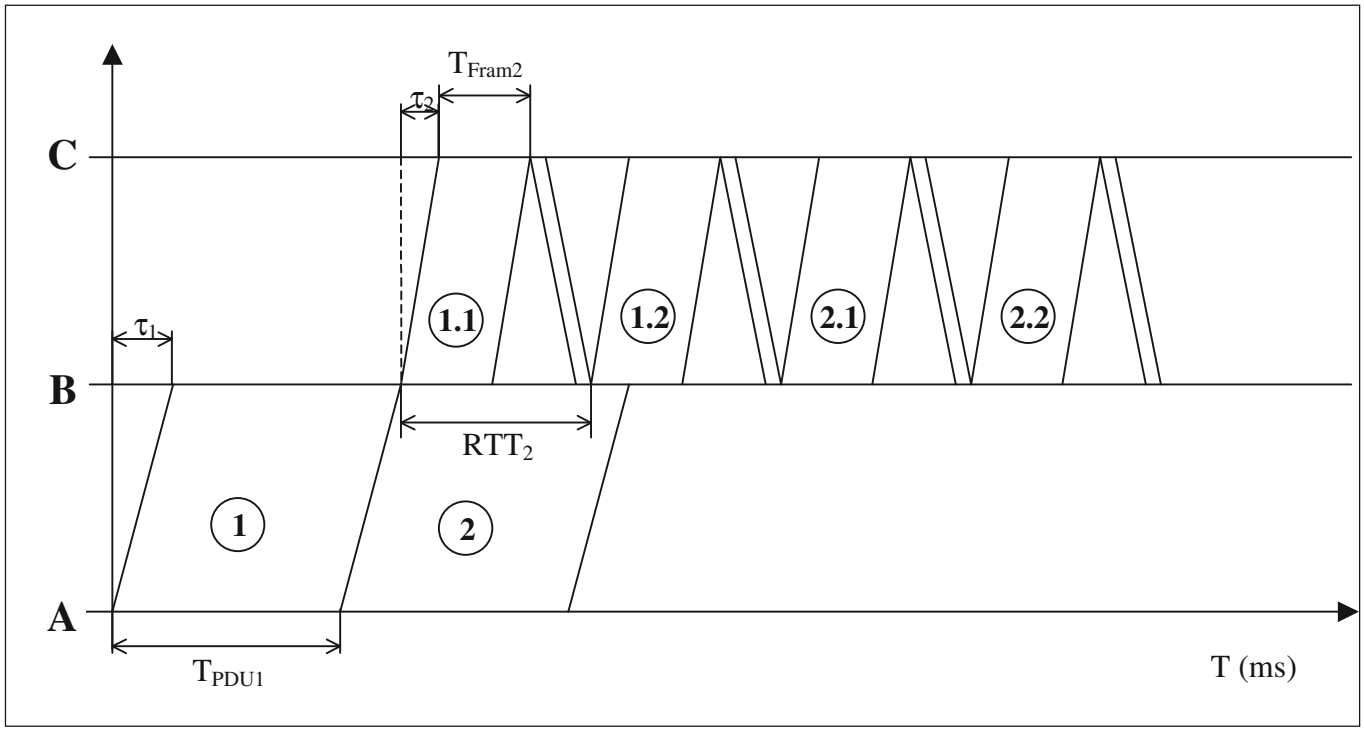
Mantenendo ferme le ipotesi di cui al punto 1 e supponendo di far variare n (DLC2) da 1 a infinito, calcolare la funzione $C_{sistema}(n)$ tracciandone altresì il grafico.

Soluzione

C_1 = 24000 bps = 3000 Byteps $\quad\quad$ τ_1 = 150 ms = 0,15 s
C_2 = 32000 bps = 4000 Byteps $\quad\quad$ τ_2 = 100 ms = 0,1 s

Il livello Network deve eseguire la frammentazione nei confronti del DLC2, secondo il seguente schema:





$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{1700 + 100}{3000} = 0{,}6s$$

$$T_{Fram2} = \frac{Payload_{fram} + H_{DLC2}}{C_2} = \frac{900 + 100}{4000} = 0{,}25s$$

$$RTT_2 = 2\tau_2 + \frac{Payload_{DLC2} + 2H_{DLC2}}{C_2} = 2 \cdot 0{,}1 + \frac{900 + (2 \cdot 100)}{4000} = 0{,}475s$$

$T_{PDU2} = 2 \cdot RTT_2 = 2 \cdot 0{,}475 = 0{,}95s$ Tempo per il trasferimento di una PDU sul canale 2 (2 frammenti)

Siccome $T_{PDU2} > T_{PDU1}$ è il canale 2 che condiziona il ritmo di arrivo delle PDU;

$$C_{sistema} = \frac{Payload_{Tr}}{2RTT_2} = \frac{1500}{0{,}475 \cdot 2} = 1578\,Byte/s$$

2° punto

C sistema dipende da C_2 quando $2RTT(C_2) \geq T_{PDU1}$, ovvero fino a che $T_{PDU1} = 2RTT$

$$T_{PDU\ 1} = 2RTT$$

$$T_{PDU\ 1} = \left(2\tau_2 + \frac{MSS\ + 2H}{C_2} \right) \cdot 2$$

$$0{,}6 = \left(0{,}2 + \frac{900\ + 200}{C_2} \right) \cdot 2$$

$$0,6 = 0,4 + \frac{1800 + 400}{C_2}$$

$$0,2 = \frac{2200}{C_2}$$

$$0,2 \cdot C_2 = 2200 \Rightarrow C_2 = 11000 \ \ Byte/s$$

- **<u>Con C_2 < 11000</u>** $\quad C_{sistema}(C_2) = \frac{Payload_{Tr}}{2RTT_2}$

$$C_{sistema}(C_2) = \frac{Payload_{Tr}}{\left(2\tau_2 + \frac{M + 2H}{C_2}\right) \cdot 2} = \frac{1500}{\left(2 \cdot 0,1 + \frac{900 + (2 \cdot 100)}{C_2}\right) \cdot 2} = \frac{1500\,C_2}{0,4C_2 + 2200}$$

$$C'_{sistema}(C_2) = \frac{1500\,(0,4C_2 + 2200) - 0,4(1500\,C_2)}{(0,4C_2 + 2200)^2} = \frac{3,3.10^6}{(0,4C_2 + 2200)^2}$$

La derivata prima è sempre positiva, quindi la funzione risulta essere sempre crescente

$$C''_{sistema}(C_2) = \frac{-3,3.10^6 (0,32C_2 + 1760)}{(0,4C_2 + 2200)^4}$$

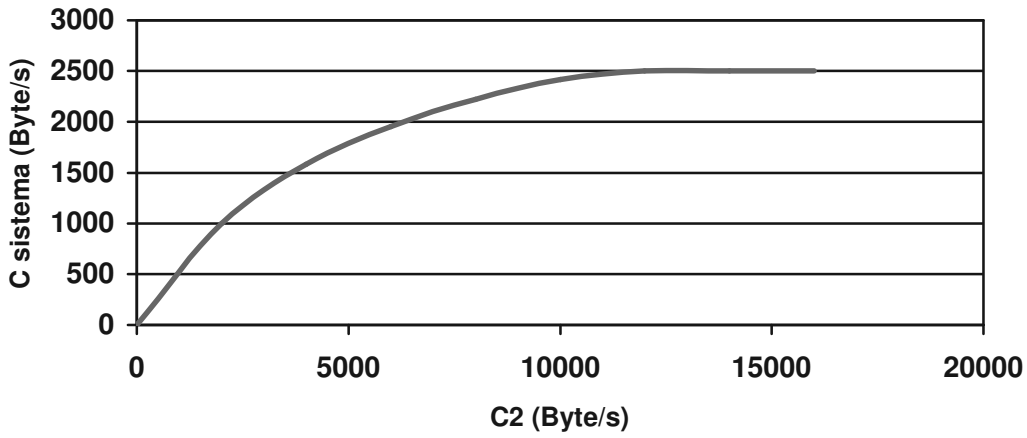
La derivata seconda è sempre negativa, quindi la funzione presenta una concavità rivolta verso il basso

Per $C_2 = 0$: $\quad C_{sistema}(0) = 0$ per cui la funzione passa per l'origine

C_2 per sua natura può assumere solo valori positivi

- **<u>Con C_2 > 11000</u>** il valore della $C_{sistema}$ rimane costante poiché non dipende più da C_2 (il canale limitante risulta essere il primo), per cui:

$$C_{sistema}(C_2) = \frac{Payload_{Tr}}{T_1} = C_1 \cdot \frac{M}{M + H_{Tot}} = 3000 \cdot \frac{1500}{1500 + 300} = 2500\,Byte/s$$



3° Punto

La funzione Csistema(n) è una funzione discreta, poiché la variabile indipendente n può assumere solo valori discreti positivi, da 1 a infinito. Al crescere della finestra (n) aumentano i frammenti che possono essere spediti in maniera consecutiva prima dell'arrivo del primo ACK.
Per questo motivo la $C_{sistema}$ cresce all'aumentare di n ed arriverà a stabilizzarsi nel momento in cui il tempo di trasferimento della finestra risulta essere maggiore di RTT.
Si procede quindi calcolando la $C_{sistema}$ aumentando progressivamente n ed analizzando la rispettiva situazione:

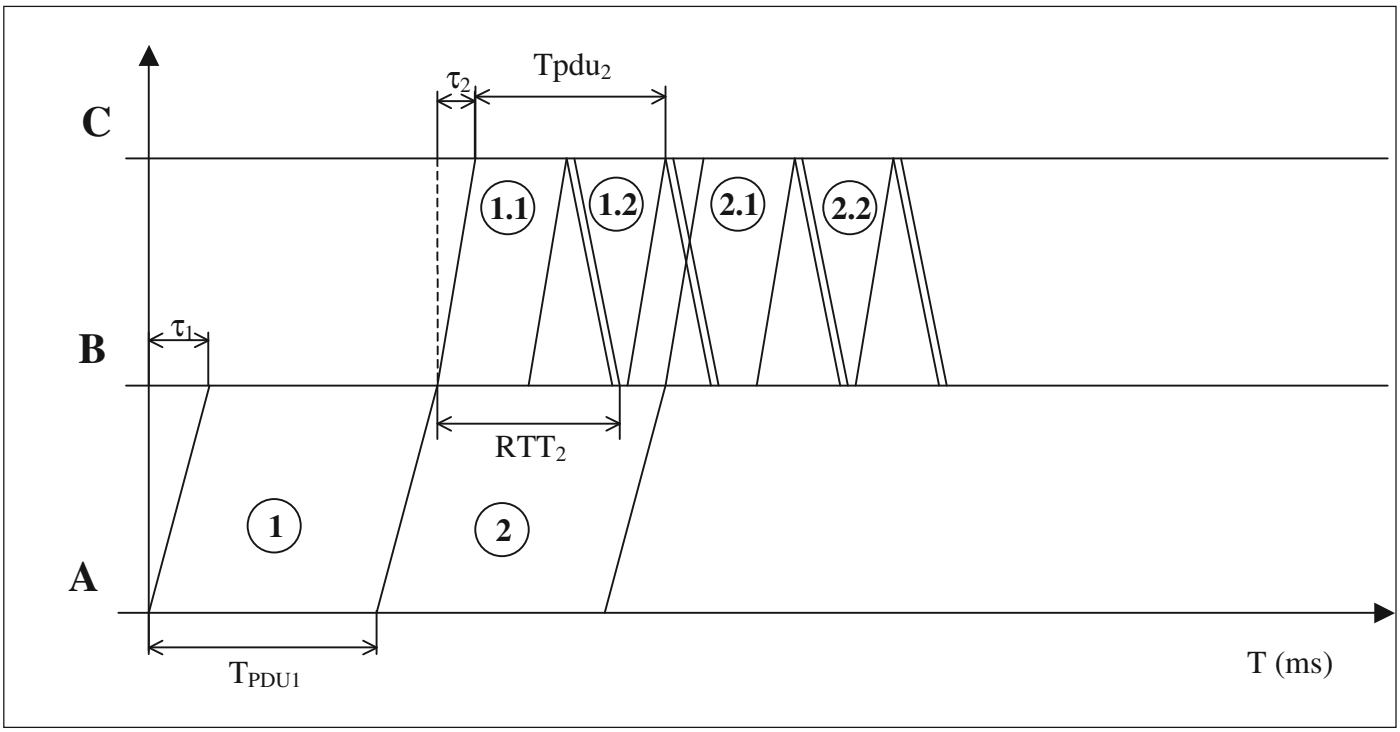
$$C_{sistema}(n=1) = \frac{Payload_{Tr}}{2RTT_2} = \frac{1500}{2 \cdot 0{,}475} = 1578\, Byte/s$$

$C_{sistema}(n=2)$:

$$T_{PDU1} = \frac{Payload_{DLC} + H_{DLC}}{C_1} = \frac{1700+100}{3000} = 0{,}6s$$

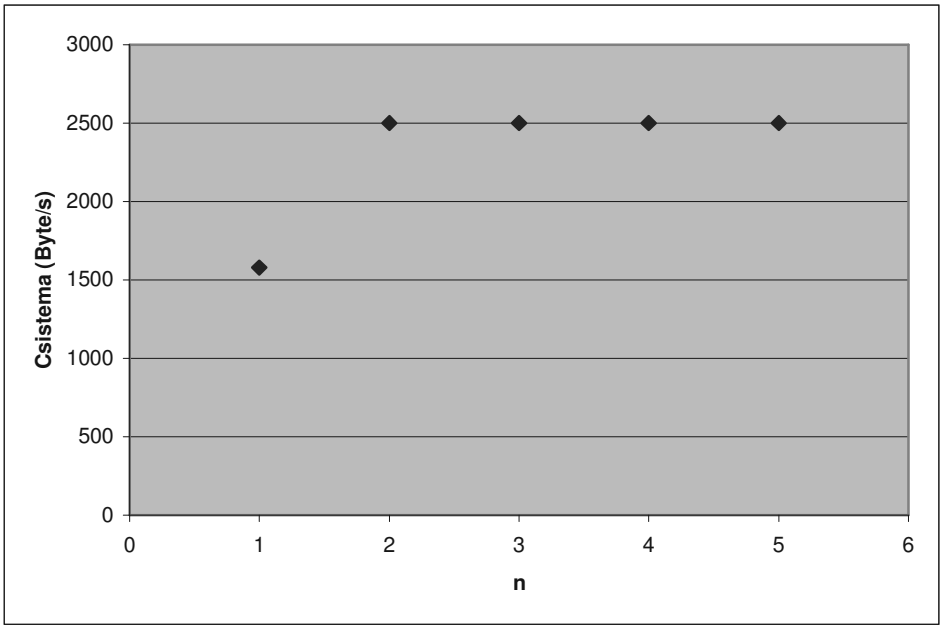
$$T_{PDU2} = T_{Window} = \frac{(Fragment_{DLC} + H_{DLC}) \cdot 2}{C_2} = \frac{(900+100) \cdot 2}{4000} = 0{,}5s$$

Il tempo di trasferimento della finestra (i due frammenti) è maggiore di RTT_2, per cui a partire da n>=2 il protocollo DLC2 si comporta come se non fosse confermato.



Essendo $T_{PDU2} < T_{PDU1}$ si può concludere che il canale che condiziona la $C_{sistema}$ è il primo. Osservando lo schema temporale si nota inoltre che all'aumentare di n (dopo il valore "2") la situazione non cambia.
Il nodo B non sfrutterà mai una finestra maggiore di 2 poiché deve rimanere in attesa dell'arrivo del pacchetto dal nodo A.
Si può concludere quindi che con n>=2 lo schema temporale rimane costante e uguale a quello sopra indicato, per cui

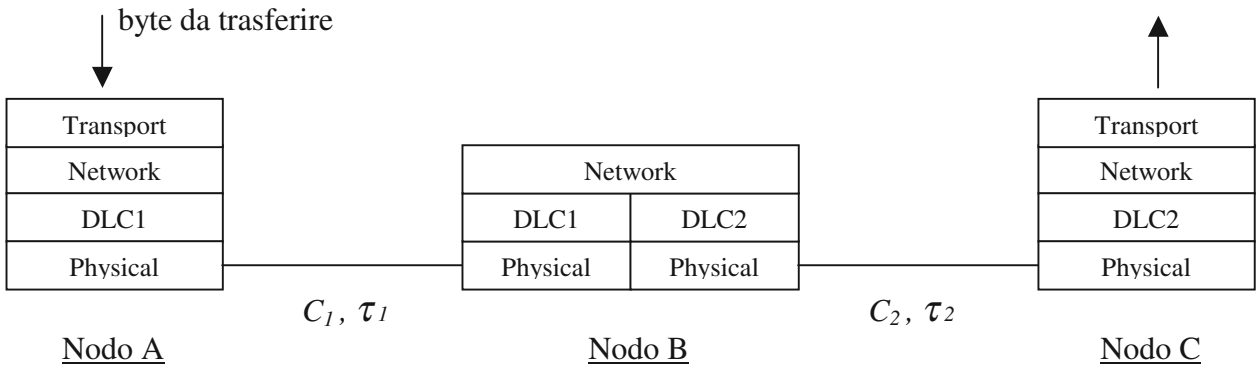
$$C_{sistema}(n > 2) = \frac{Payload_{Tr}}{T_1} = \frac{1500}{0{,}6} = 2500\ Byte/s$$



Il grafico della funzione $C_{sistema}(n)$

Esercizio 6 *(Appello del 10/02/2004)*

Sia data la rete indicata in figura (il sistema è privo di errori), nella quale il nodo B commuta i pacchetti a livello 3 in un tempo trascurabile con modalità *store-and-forward*. Tutti i nodi dispongono di buffer di dimensione infinita.



Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

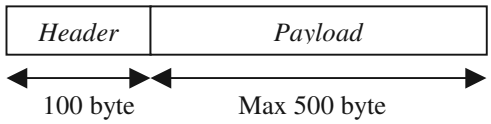
$C_1 = 32.000$ *bps* $\quad \tau_1 = 150$ *ms*

$C_2 = 24.000$ *bps* $\quad \tau_2 = 100$ *ms*

Caratteristiche dei protocolli di comunicazione:

DLC1 utilizza un protocollo confermato di tipo *Go-Back-n* (come al solito ipotizzare che l'entità ricevente generi una PDU-ACK per ogni PDU-DATI corretta ricevuta), con *n* specificato più avanti

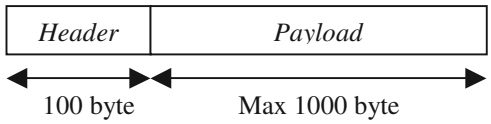
PDU-DATI di DLC1



PDU-ACK di DLC1: è costituita dalla sola *header* di 100 byte

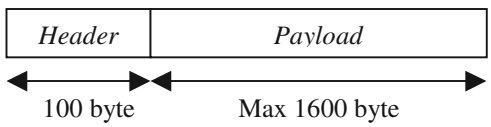
DLC2 utilizza un protocollo non confermato

PDU-DATI di DLC2



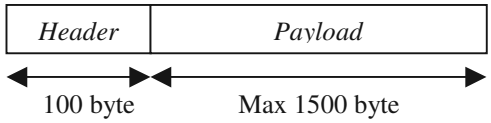
Network utilizza un protocollo non confermato, il quale prevede, quando serve, la funzione di frammentazione (come al solito ipotizzare che l'operazione di ricomposizione della PDU frammentata venga svolta solo sul destinatario finale C).

PDU-DATI di Network



Transport utilizza un protocollo non confermato

PDU-DATI di Transport



Domande:
(Disegnare tutti gli schemi temporali; spiegare **sempre** ed in dettaglio ogni passo/assunzione fatti)

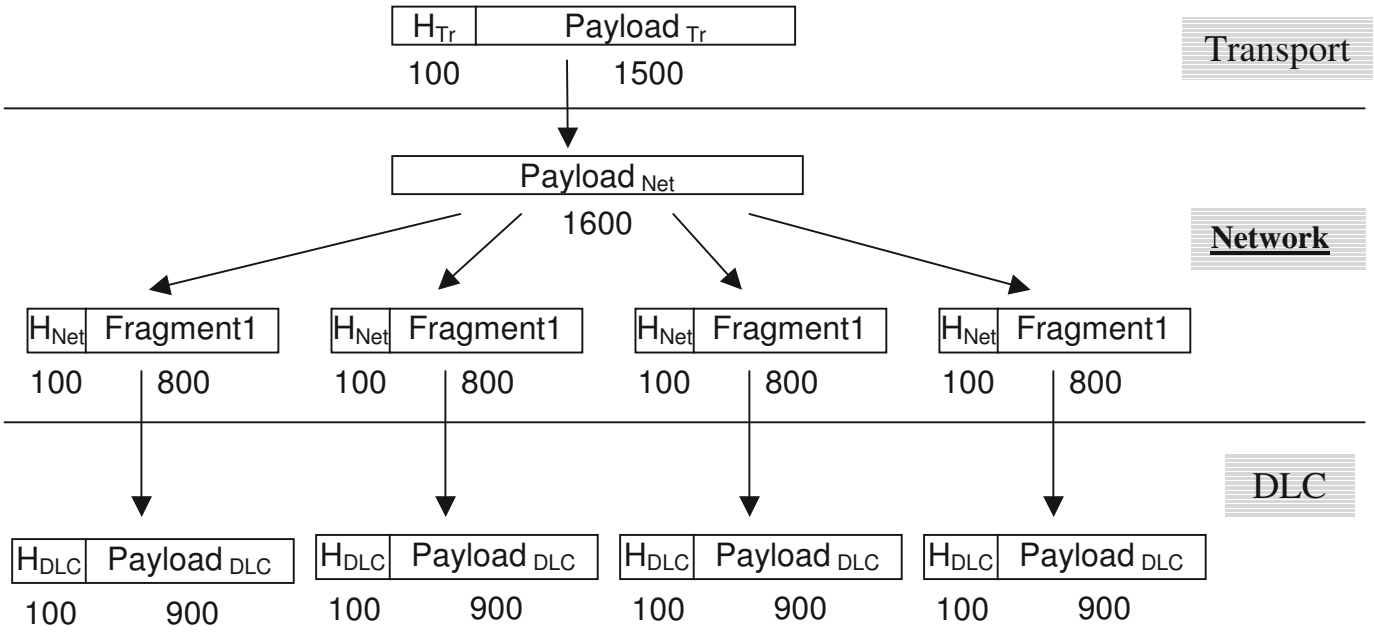
1. Nel caso in cui *n=1* (DLC1) calcolare:
   a. la capacità $C_{sistema}$ sperimentata al di sopra del livello *Transport* in un trasferimento da A a C.
   b. il ritardo $\tau_{end\text{-}to\text{-}end}$ da A a C sperimentato al di sopra del livello *Transport* da un messaggio di lunghezza pari a 1500 byte (si ipotizzi che sulla rete venga trasferito solo tale messaggio).

Calcolare $C_{sistema}$ e $\tau_{end\text{-}to\text{-}end}$ richiesti ai punti (a) e (b) della precedente domanda, nel caso in cui *n=2* (DLC1), calcolandone altresì la variazione percentuale (rispetto ai valori calcolati alla domanda precedente).

Soluzione

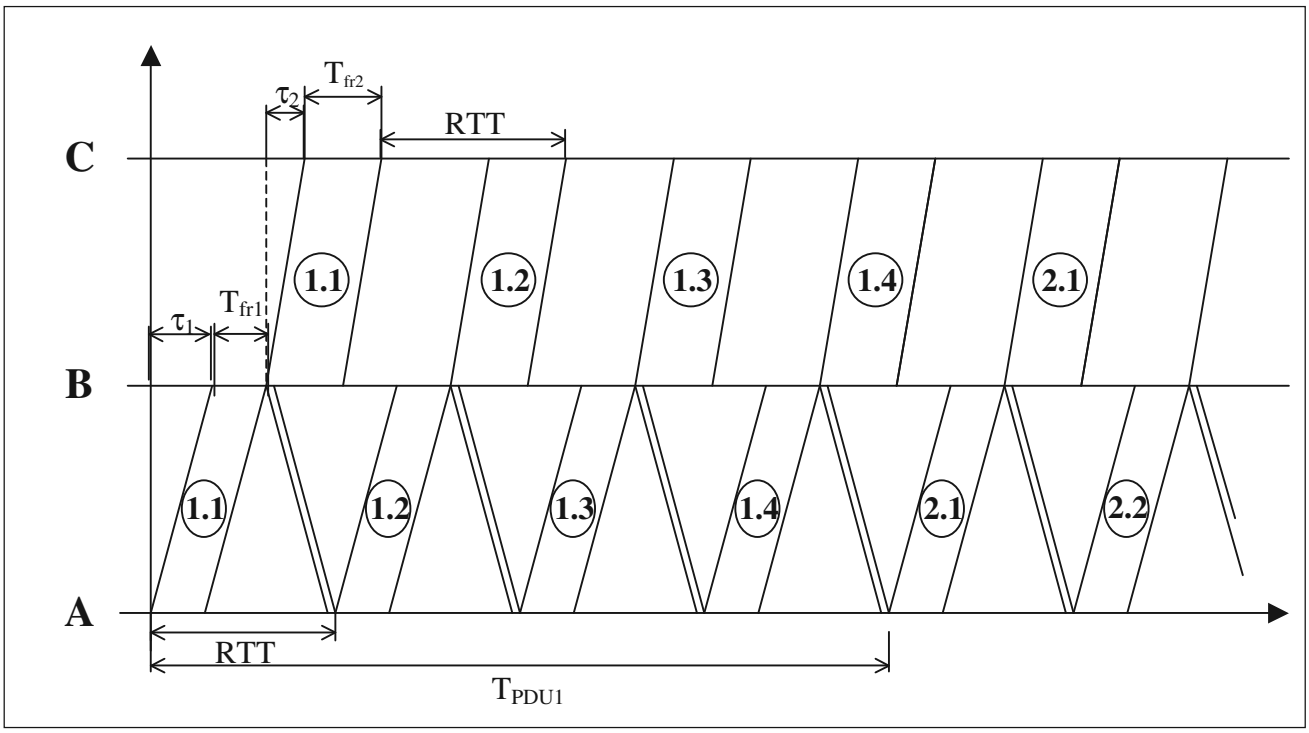
C_1 = 24000 bps = 3000 byte/s τ_1 = 150 ms = 0,15 s
C_2 = 32000 bps = 4000 byte/s τ_2 = 100 ms = 0,1 s

Il livello Network deve eseguire la frammentazione nei confronti del DLC1, secondo il seguente schema:



Punto 1:

In questo primo caso DLC1 utilizza un protocollo confermato di tipo Go-Back-n con n pari ad 1, funzionando peranto di fatto come un protocollo di tipo Stop-and-Wait. Sapendo inoltre che il protocollo utilizato da DLC2 è di tipo non-confermato, possiamo disegnare lo schema temporale sotto riportato.



Dovendo calcolare la C del sistema sperimentata al di sopra del livello Transport, iniziamo con il calcolare i tempi impiegati per trasmettere un frammento rispettivamente sul primo e sul secondo canale e quindi l'RTT relativo alla trasmissione lungo il canale 1 di un frammento :

$$T_{fram1} = \frac{Payload_{fram} + H_{DLC}}{C_1} = \frac{500+100}{4000} = 0{,}15s$$

$$T_{fram2} = \frac{Payload_{fram} + H_{DLC}}{C_2} = \frac{500+100}{3000} = 0{,}2s$$

$$RTT_1 = 2\tau_1 + \frac{Payload_{fram} + 2H_{DLC}}{C_1} = 2 \cdot 0{,}15 + \frac{500 + (2 \cdot 100)}{4000} = 0{,}475s$$

A questo punto possiamo calcolare i tempi totali di trasmissione di una PDU (composta come visto da 4 frammenti) sul primo e sul secondo canale e confrontarli quindi tra loro…

$$T_{PDU1} = 4 \cdot RTT_1 = 4 \cdot 0{,}475 = 1{,}9s$$

$$T_{PDU2} = 4 \cdot T_{fram2} = 4 \cdot 0{,}2 = 0{,}8s$$

Siccome $T_{PDU1} > T_{PDU2}$, sarà il canale 1 a condizionare il ritmo di arrivo delle PDU e la capacità del sistema sperimentata al di sopra del livello Transport in un trasferimento da A a C sarà pari a:

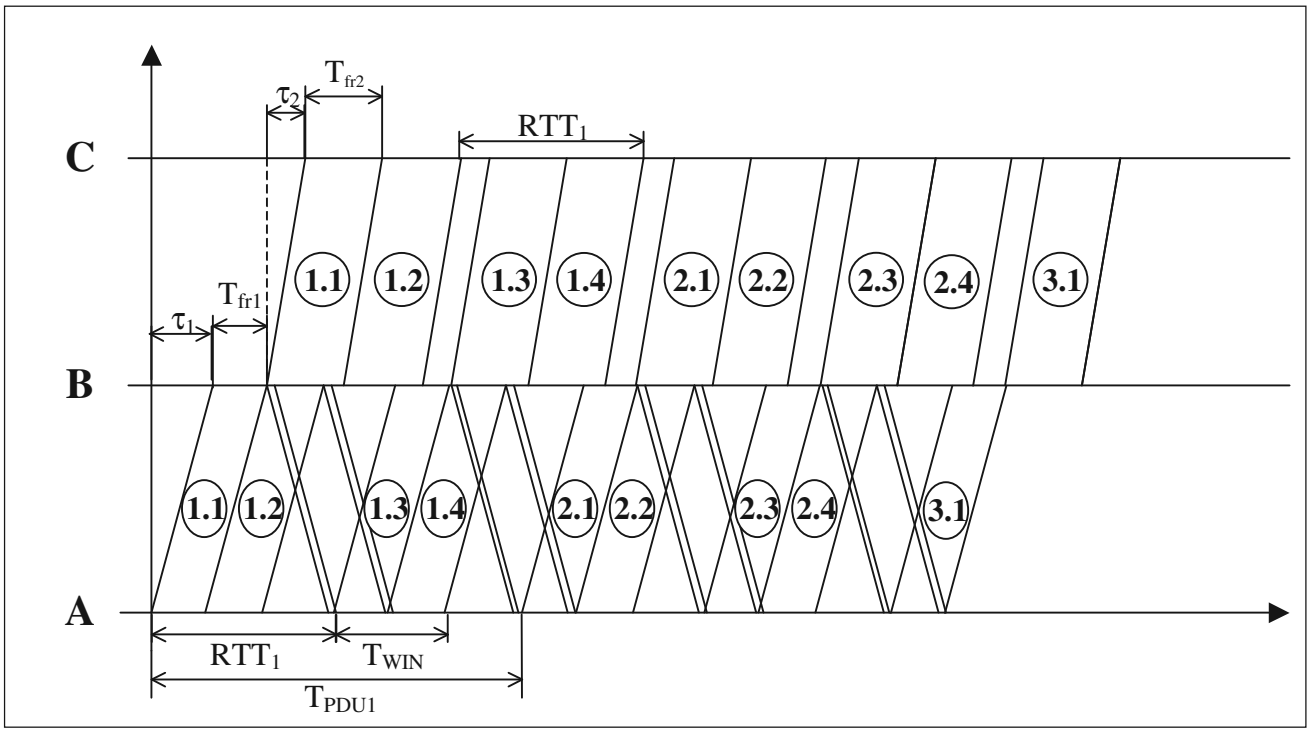
$$C_{sistema} = \frac{PDU_{TRANSPORT}}{T_{PDU1}} = \frac{PDU_{TRANSPORT}}{4RTT_1} = \frac{1500}{0{,}475 \cdot 4} = 789\, byte/s$$

Il ritardo $\tau_{end\text{-}to\text{-}end}$ da A a C sperimentato al di sopra del livello Transport da un messaggio di 1500 byte altro non è che il tempo che intercorre dall'istante in cui parte dal nodo A il primo frammento della PDU-dati all'istante in cui l'ultimo frammento che compone la stessa PDU-dati giunge sul nodo destinatario. Osservando quindi lo schema temporale si ricava che:

$$\tau_{end-to-end} = \tau_1 + T_{fram1} + \tau_2 + T_{fram2} + 3RTT_1 = 0{,}15 + 0{,}15 + 0{,}1 + 0{,}2 + 3 \cdot 0{,}475 = 2{,}025s$$

Punto 2:

Nel secondo caso DLC1 utilizza un protocollo confermato di tipo Go-Back-n con n=2 mentre DLC2 è sempre di tipo non-confermato. Lo schema temporale risulta quindi differente da quello precedente:



Poichè sul canale 1 è utilizzato un protocollo confermato "a finestra" di tipo Go-Back-n, occorre per prima cosa confrontare il tempo di apertura della finestra di trasmissione T_{WIN} con il RTT e dedurre il tipo di trasmissione a regime (ovviamente considerando il sistema privo di errori):

$$T_{fram1} = \frac{Payload_{fram} + H_{DLC}}{C_1} = \frac{500+100}{4000} = 0{,}15s$$

$$T_{WIN} = 2 \cdot \frac{Payload_{fram} + H_{DLC}}{C_1} = 2 \cdot \frac{500+100}{4000} = 0{,}3s$$

$$RTT_1 = 2\tau_1 + \frac{Payload_{fram} + 2H_{DLC}}{C_1} = 2 \cdot 0{,}15 + \frac{500 + (2 \cdot 100)}{4000} = 0{,}475s$$

Essendo $T_{WIN} < RTT_1$, sul primo canale non si instaurerà una trasmissione continua bensì verranno spediti 2 frammenti ogni RTT.

Tenendo sempre a mente che la PDU-Dati presa in considerazione viene divisa in 4 frammenti, possiamo calcolare i nuovi tempi totali di trasmissione di una PDU sul primo e sul secondo canale.

$$T_{PDU1} = 2 \cdot RTT_1 = 2 \cdot 0{,}475 = 0{,}95s$$

$$T_{fram2} = \frac{Payload_{fram} + H_{DLC}}{C_2} = \frac{500+100}{3000} = 0{,}2s$$

$$T_{PDU2} = 4 \cdot T_{fram2} = 4 \cdot 0{,}2 = 0{,}8s$$

Come nel caso precedente, $T_{PDU1} > T_{PDU2}$ e pertanto sarà sempre il canale 1 a condizionare il ritmo di arrivo delle PDU. La capacità del sistema sperimentata al di sopra del livello Transport in un trasferimento da A a C sarà pari a:

$$C_{sistema} = \frac{PDU_{TRANSPORT}}{T_{PDU1}} = \frac{PDU_{TRANSPORT}}{2RTT_1} = \frac{1500}{0{,}475 \cdot 2} = 1579\, {byte}/{s}$$

Sempre aiutandosi con lo schema temporale, si trova inoltre che il ritardo $\tau_{end\text{-}to\text{-}end}$ da A a C sperimentato al di sopra del livello Transport da un messaggio di 1500 byte è pari a:

$$\tau_{end-to-end} = \tau_1 + T_{fram1} + \tau_2 + 2 \cdot T_{fram2} + RTT_1 = 0{,}15 + 0{,}15 + 0{,}1 + 2 \cdot 0{,}2 + 0{,}475 = 1{,}275s$$

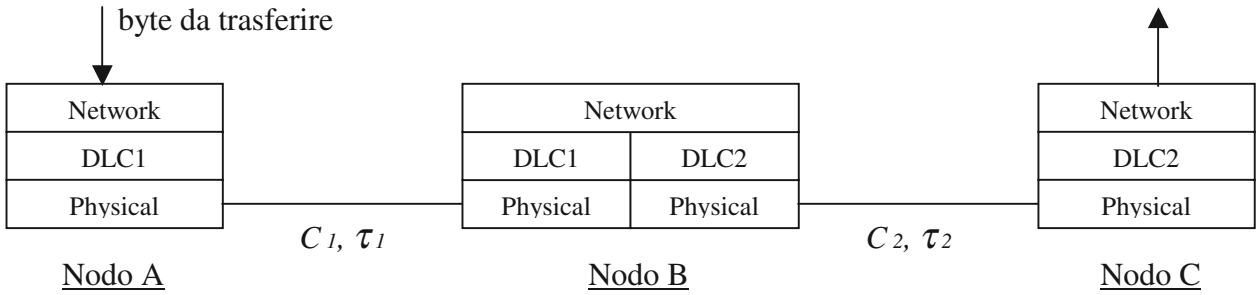
Volendo calcolare le variazioni percentuali di $C_{sistema}$ e del ritardo $\tau_{end\text{-}to\text{-}end}$ rispetto ai valori calcolati precedentemente, si trova che:

$$\Delta C_{sistema}\% = \frac{C_{sistema2} - C_{sistema1}}{C_{sistema1}} \cdot 100 = \frac{1579 - 789}{789} \cdot 100 = +100\%$$

$$\Delta \tau_{endtoend}\% = \frac{\tau_{endtoend2} - \tau_{endtoend1}}{\tau_{endtoend1}} \cdot 100 = \frac{1{,}275 - 2{,}025}{2{,}025} \cdot 100 = -37\%$$

Esercizio 7 *(Appello del 25/02/2004)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta i pacchetti in modalità *store-and-forward* con un tempo di commutazione trascurabile.



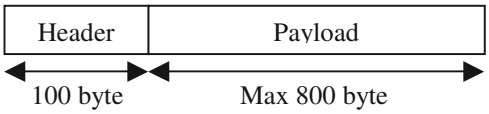
Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

$C_1 = 16.000$ bps $\quad \tau_1 = 100$ ms
$C_2 = 20.000$ bps $\quad \tau_2 = 100$ ms

Caratteristiche dei protocolli di comunicazione:

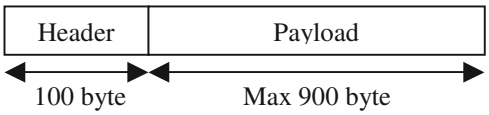
Network utilizza un protocollo non confermato, il quale prevede, quando serve, la funzione di frammentazione (come al solito ipotizzare che l'operazione di ricomposizione della PDU frammentata venga svolta solo sul destinatario finale C)

PDU-DATI di Network

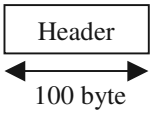


Il livello **DLC1** utilizza un protocollo confermato di tipo *Go-Back-n* (con *n* specificato nella sezione domande):

PDU-DATI di DLC1

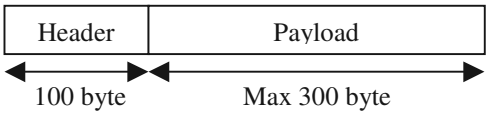


PDU-ACK di DLC1



Il livello **DLC2** utilizza un protocollo non confermato:

PDU-DATI di DLC2



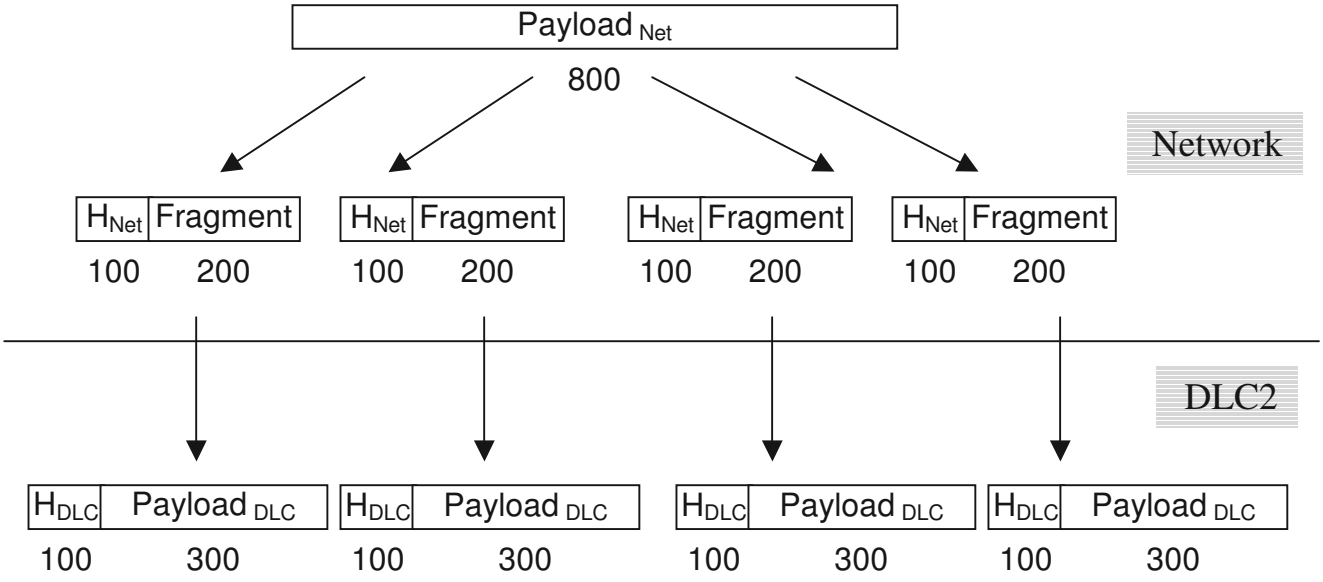
Domande:
(Disegnare sempre tutti gli schemi temporali di trasferimento dei messaggi)

Determinare la funzione capacità del sistema *Csistema(n)* sperimentata al di sopra del livello *Network*, quando è in corso un trasferimento di una sequenza illimitata di byte dal nodo A al nodo C, indicandone altresì i valori per *n=1*, *n=2*, *n=3*.

Soluzione

C_1 = 16000 bps = 2000 Byte/s τ_1 = 100 ms = 0,1 s
C_2 = 20000 bps = 2500 Byte/s τ_2 = 100 ms = 0,1 s

Il livello Network deve eseguire la frammentazione nei confronti del DLC2, secondo il seguente schema:



$$T_{PDU1} = \frac{Payload_{DLC1} + H_{DLC1}}{C_1} = \frac{800+200}{2000} = 0{,}5s$$

$$T_{ACK} = \frac{Ack}{C_1} = \frac{100}{2000} = 0{,}05s$$

$$T_{FRAM2} = \frac{Payload_{DLC2} + H_{DLC2}}{C_2} = \frac{300+100}{2500} = 0{,}16s$$

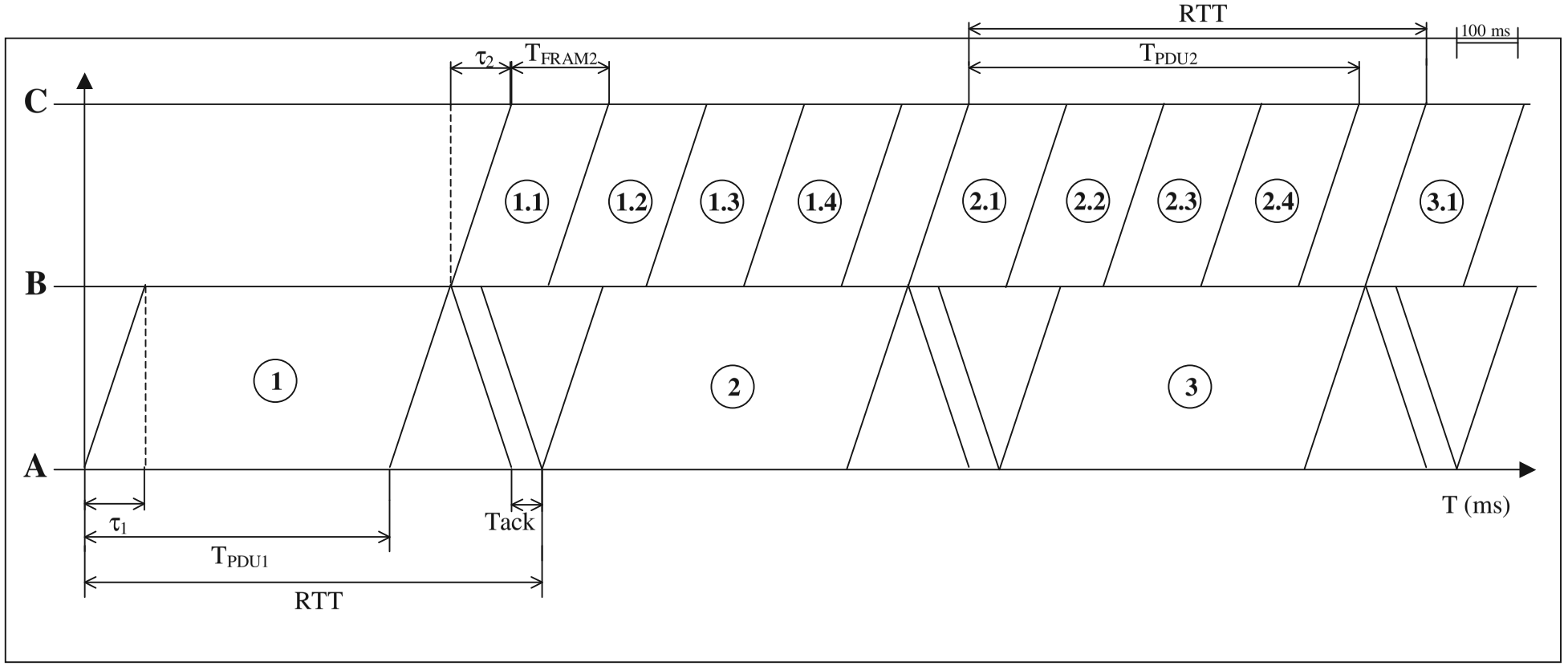
$$T_{PDU2} = T_{FRAM2}.4 = 0{,}16.4 = 0{,}64s$$

$$RTT = 2\tau_1 + T_{PDU1} + T_{ACK} = 2 \cdot 0{,}1 + 0{,}5 + 0{,}05 = 0{,}75s$$

Caso n=1:

Per n=1 il protocollo Go-Back-1 si comporta come se fosse un protocollo Stop&Wait.
Inoltre RTT>T_{PDU2}, ovvero lo strozzante è il canale A-B, perciò la capacità del sistema al di sopra del livello Network dipende dal RTT.

$$C_{Sistema} = \frac{Payload_{Network}}{RTT} = \frac{800}{0,75} = 1066,66 Byte/s$$

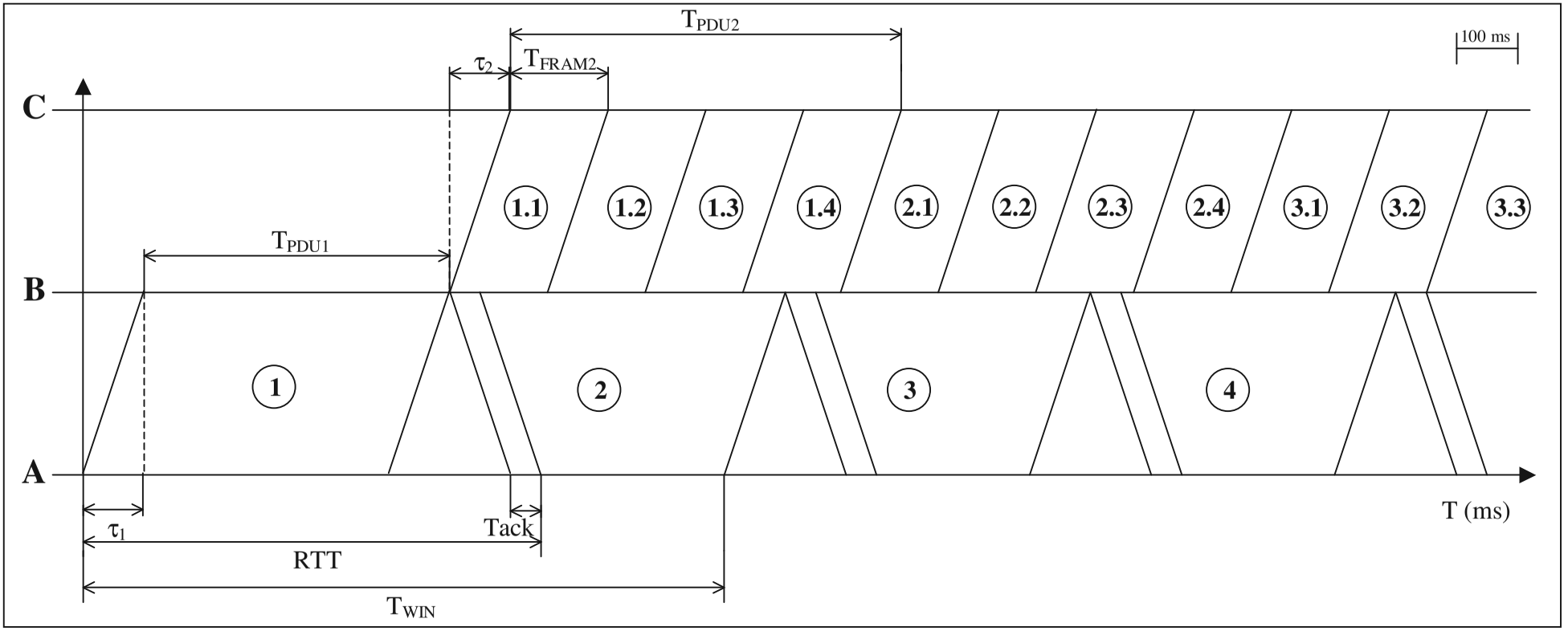


Caso n=2:

$$T_{WINDOW} = Payload_{DLC1}.2 = 0{,}5.2 = 1s$$

Per n=2 il T_{WINDOW}>RTT. Questo significa che il protocollo Go-Back-2 si comporta con la caratteristica di un normale trasferimento in continua non confermato, quindi i tempi di trasmissione delle Ack sono ininfluenti sul calcolo prestazionale. Il canale strozzante in questo caso è B-C perché T_{PDU2}>T_{PDU1}. La capacità del sistema al di sopra del livello Network dipenderà perciò da T_{PDU2}.

$$C_{Sistema} = \frac{Payload_{Network}}{T_{PDU\,2}} = \frac{800}{0{,}64} = 1250Byte/s$$

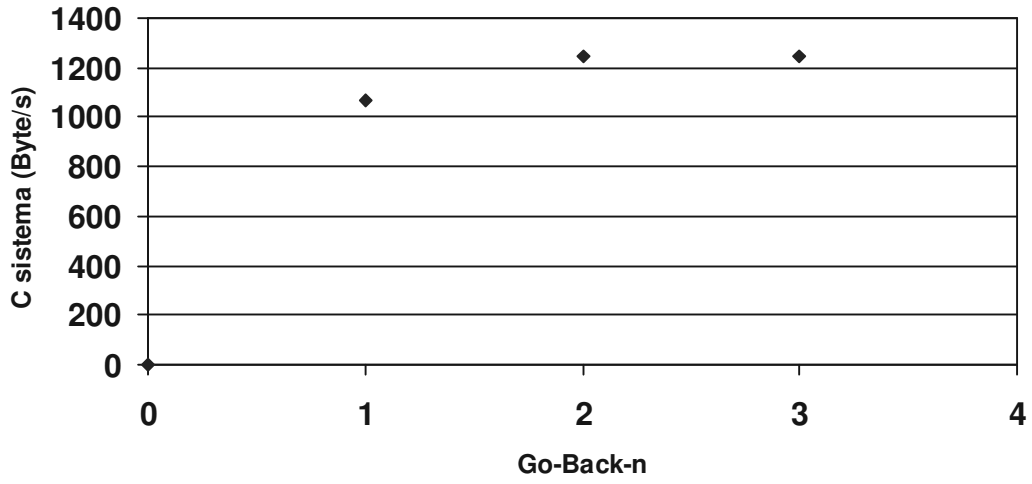


Caso n=3:

Per n=3 la capacità del sistema al di sopra del livello Network non cambia perché, come prima, T_{WINDOW}>RTT. Il protocollo Go-Back-3 si comporta quindi come un non-confermato. Come nel caso, precedente la $C_{Sistema}$ = 1250 Byte/s.

A questo punto possiamo visualizzare l'andamento della $C_{Sistema}$ al variare dell'apertura della finestra di trasmissione:

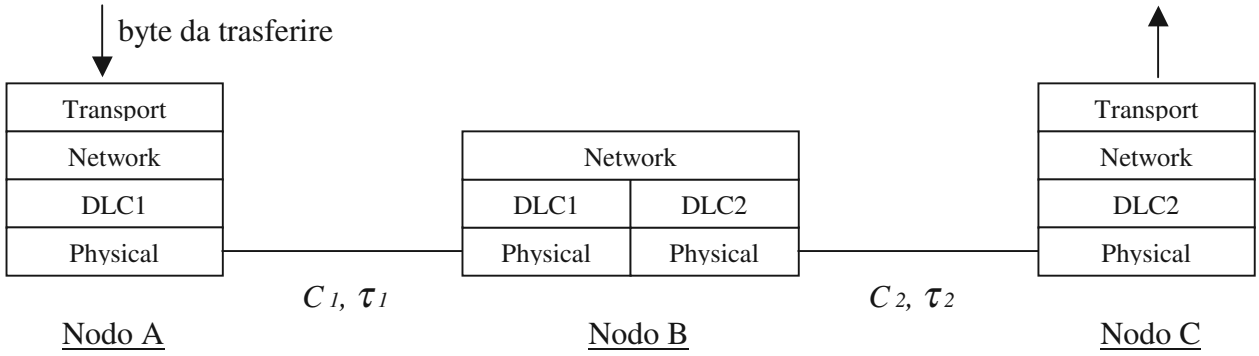
n	0	1	2	3	…
C sistema (Byte/s)	0	1067	1250	1250	1250



Andamento della $C_{Sistema}$ in funzione di n (a valori discreti).

Esercizio 8 *(1° Itinere del 30/04/2002)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B commuta i pacchetti in modalità *store-and-forward* con $\tau_{forwarding} = 0$.



Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

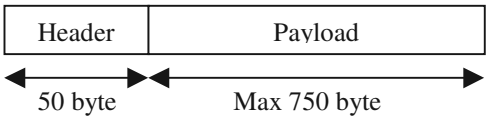
$C_1 = 32$ Kbps $\quad \tau_1 = 100$ ms

$C_2 = 16$ Kbps $\quad \tau_2 = 50$ ms

Caratteristiche dei protocolli di comunicazione:

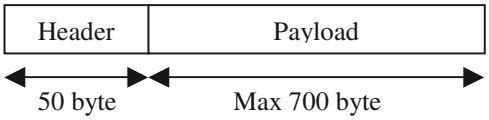
DLC1 e DLC2 utilizzano lo stesso protocollo, il quale è non confermato:

PDU-DATI di DLC1 e 2



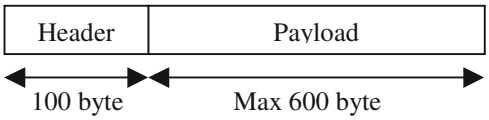
Network utilizza un protocollo non confermato:

PDU-DATI di Network

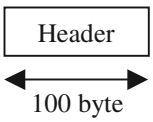


Transport utilizza un protocollo confermato di tipo *Stop-and-Wait*:

PDU-DATI di Transport



PDU-ACK di Transport



Domande:

1) Calcolare la capacità del sistema sperimentata al di sopra del livello *Transport*, nonché l'utilizzo medio dei link, quando è in corso un trasferimento di una sequenza illimitata di byte dal nodo A al nodo C.
2) Calcolare la capacità del sistema sperimentata al di sopra del livello *Transport*, quando è in corso un trasferimento di una sequenza di byte dal nodo A al nodo C, nell'ipotesi in cui:
   - La lunghezza massima del Payload di DLC2 è pari a 250 byte (le lunghezze massime del Payload degli altri protocolli rimangono invariate).

- Il protocollo di livello *Network* supporta la frammentazione delle PDU (ricordare lo schema della frammentazione visto a lezione nella parte "*Le funzioni dei layer*").

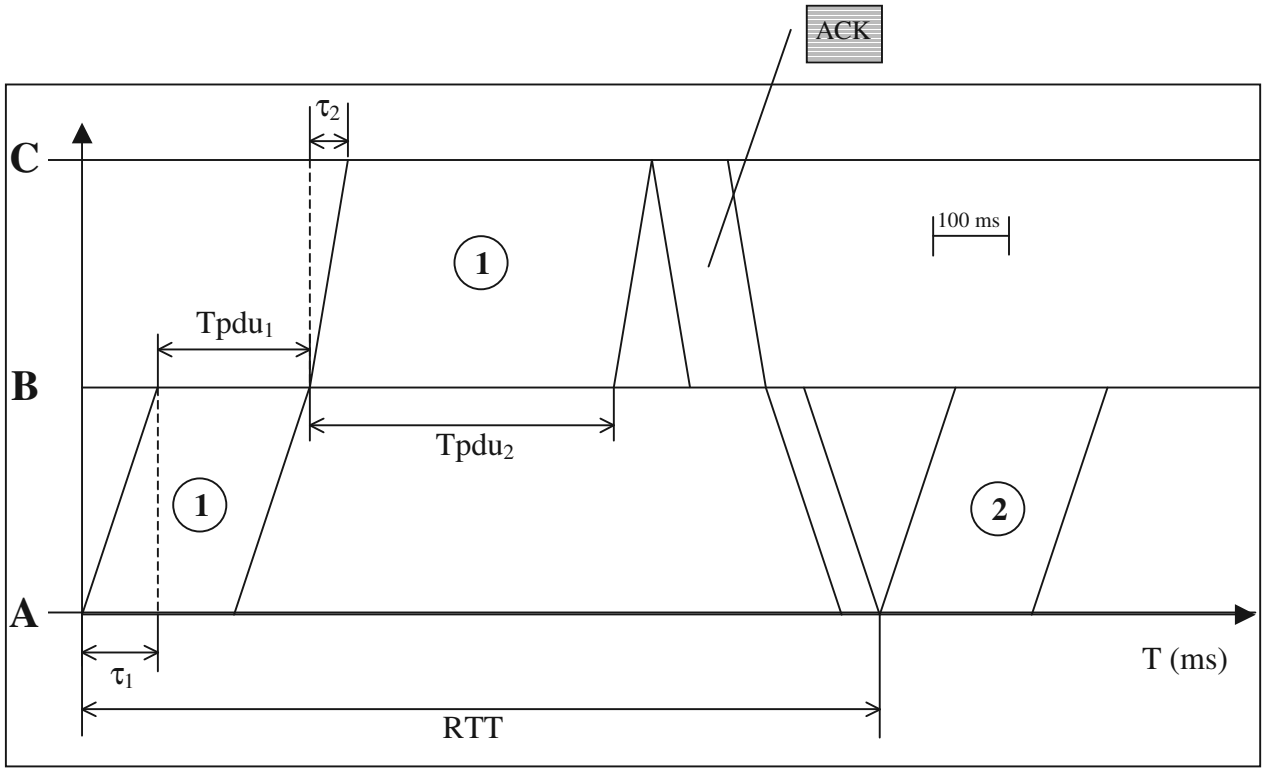
3) Con riferimento alla domanda 1, cosa cambierebbe (capacità e utilizzo dei link) nel caso in cui i canali di trasmissione fossero *half-duplex* ? Giustificare sempre le risposte.

Soluzione

C_1 = 32 Kbps = 4000 $Byte/s$ τ_1 = 100 ms = 0,1 s

C_2 = 16 Kbps = 2000 $Byte/s$ τ_2 = 50 ms = 0,05 s

1° punto



I tempi per il trasferimento di una PDU-DATI rispettivamente sui canali 1 e 2 sono:

$$T_{PDU1} = \frac{Payload_{DLC} + H_{DLC}}{C_1} = \frac{750+50}{4000} = 0{,}2s$$

$$T_{PDU2} = \frac{Payload_{DLC} + H_{DLC}}{C_2} = \frac{750+50}{2000} = 0{,}4s$$

Allo stesso modo, i tempi necessari per il trasferimento di una PDU-ACK di conferma lungo i due canali sono:

$$T_{ACK1} = \frac{H_{TRANSPORT} + H_{NETWORK} + H_{DLC}}{C_1} = \frac{100+50+50}{4000} = 0{,}05s$$

$$T_{ACK2} = \frac{H_{TRANSPORT} + H_{NETWORK} + H_{DLC}}{C_2} = \frac{100+50+50}{2000} = 0{,}1s$$

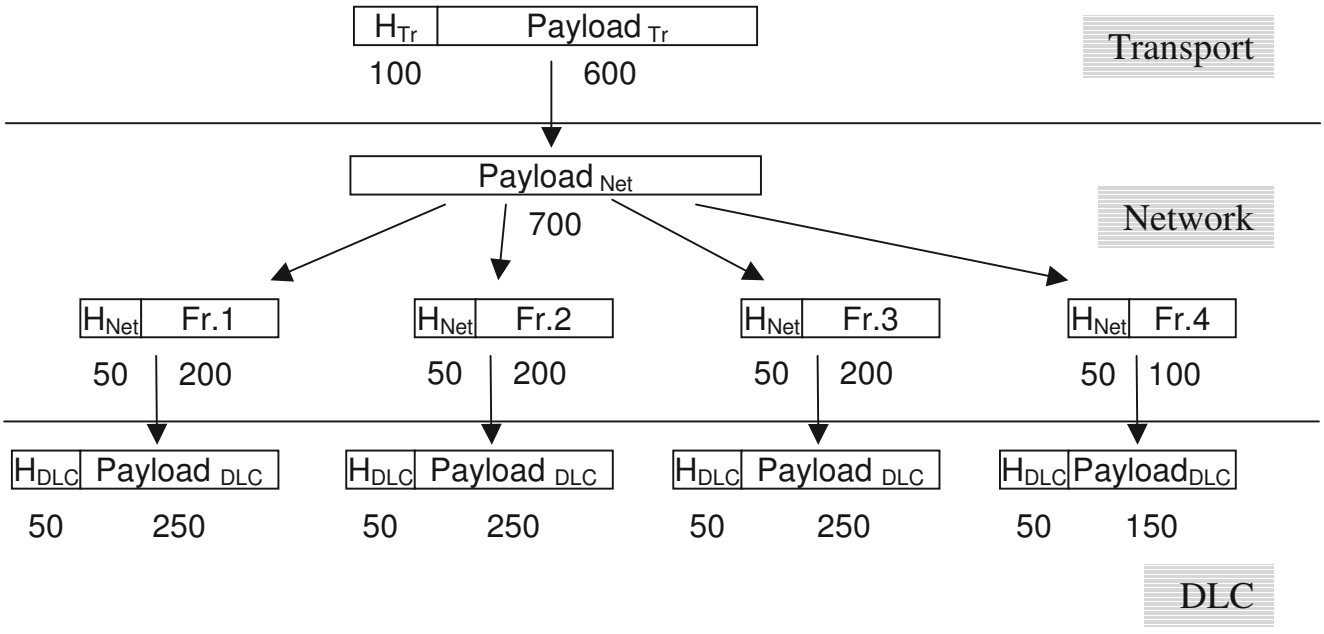
Basandosi sul grafico temporale, a questo punto è facile calcolare il RTT e, di conseguenza, la capacità del sistema vista al di sopra del livello Transport.

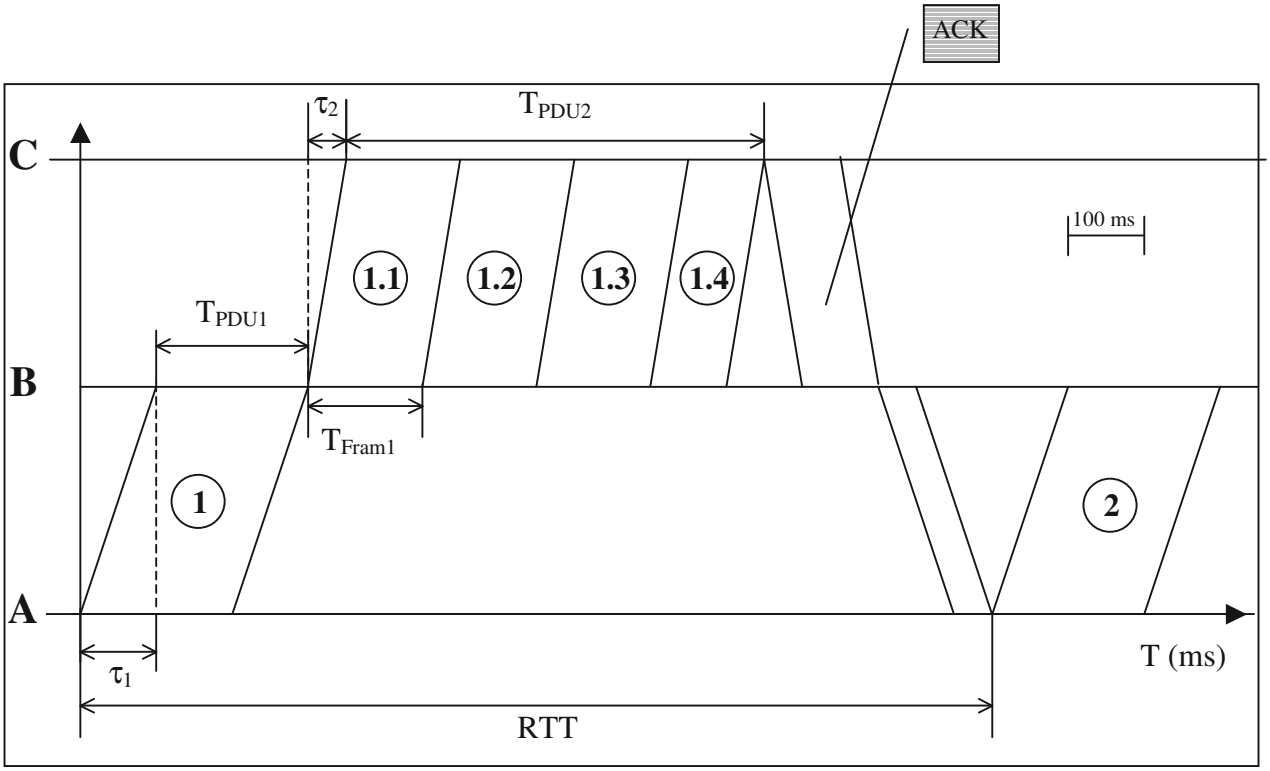
$$RTT = \tau_1 + T_{PDU1} + \tau_2 + T_{PDU2} + \tau_2 + T_{ACK2} + \tau_1 + T_{ACK1} =$$
$$= 0{,}1 + 0{,}2 + 0{,}05 + 0{,}4 + 0{,}05 + 0{,}1 + 0{,}1 + 0{,}05 = 1{,}05s$$

$$C_{sistema} = \frac{Payload_{TRANSPORT}}{RTT} = \frac{600}{1{,}05} = 571 Byte/s$$

2° punto

Il livello Network deve eseguire la frammentazione nei confronti del DLC2, secondo il seguente schema:





I tempi per il trasferimento sul canale 2 dei diversi frammenti sono:

$$T_{Fram1} = T_{Fram2} = T_{Fram3} = \frac{Payload_{Fram1} + H_{DLC}}{C_2} = \frac{250+50}{2000} = 0{,}15s$$ Tempo per 1° frammento (1.1)

$$T_{Fram4} = \frac{Payload_{Fram4} + H_{DLC}}{C_2} = \frac{150+50}{2000} = 0{,}1s$$ Tempo per 4° frammento (1.4)

T_{PDU2} è formato dai 3 frammenti da 300 byte più il frammento da 200 byte:

$$T_{PDU2} = 3 \cdot T_{Fram1} + T_{Fram4} = 3 \cdot 0{,}15 + 0{,}1 = 0{,}55s$$

$$RTT = \tau_1 + T_{PDU1} + \tau_2 + T_{PDU2} + \tau_2 + T_{ACK2} + \tau_1 + T_{ACK1} =$$
$$= 0{,}1 + 0{,}2 + 0{,}05 + 0{,}55 + 0{,}05 + 0{,}1 + 0{,}1 + 0{,}05 = 1{,}2s$$

$$C_{sistema} = \frac{Payload_{TRANSPORT}}{RTT} = \frac{600}{1{,}2} = 500\,Byte/s$$

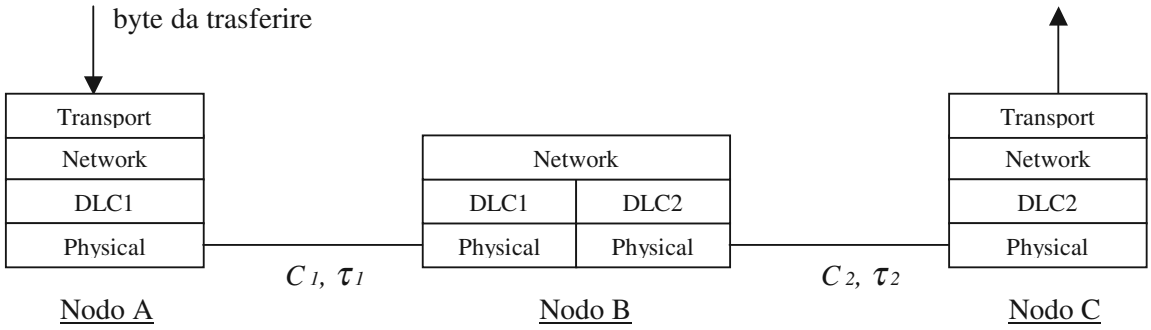
Il nuovo $C_{sistema}$ è leggermente minore a causa dell'overhead di incapsulamento.

3° punto

Se i canali di trasmissione fossero *half-duplex* non cambierebbe nulla perché il protocollo confermato è di tipo *Stop-and-Wait* e quindi non si ha una sovrapposizione tra le PDU-DATI e le PDU-ACK.

Esercizio 9 *(Appello del 25/02/2003)*

Sia data la rete indicata in figura (il sistema è privo di errori) dove il nodo B, avente buffer di dimensione infinita, commuta le PDU di livello 3 con modalità *store-and-forward* con $\tau_{forwarding} = 0$.



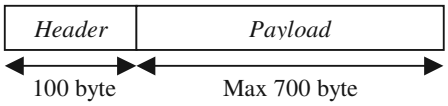
Caratteristiche dei canali di trasmissione (entrambi *full-duplex*):

$C_1 = 16000$ *bps* $\quad \tau_1 = 75$ *ms*
C_2 = *da determinare* $\quad \tau_2 = 200$ *ms*

Caratteristiche dei protocolli di comunicazione:

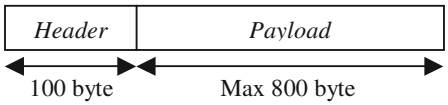
Il livello **Transport** utilizza un protocollo **non** confermato:

PDU-DATI:



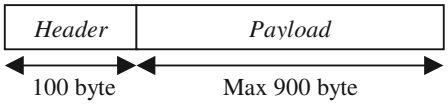
Il livello **Network** utilizza un protocollo **non** confermato:

PDU-DATI:



I livelli **DLC1** e **DLC2** utilizzano un protocollo **non** confermato

PDU-DATI:



Domande:
(Disegnare gli schemi temporali di trasferimento dei messaggi giustificando sempre ogni espressione analitica riportata)

1. Calcolare il bit-rate C_2 affinché la capacità del sistema ($C_{SISTEMA}$) sperimentata al di sopra del livello *Transport* sia pari a 350 Byte/s.

2. Utilizzando il valore C_2 calcolato al punto 1, supponendo che la dimensione massima del *Payload* del protocollo DLC1 sia pari a 500 byte anziché 900 byte e sapendo che il protocollo di livello *Network* supporta la frammentazione (nello stile IPv4), calcolare la capacità del sistema ($C_{SISTEMA}$) sperimentata al di sopra del livello *Transport*.

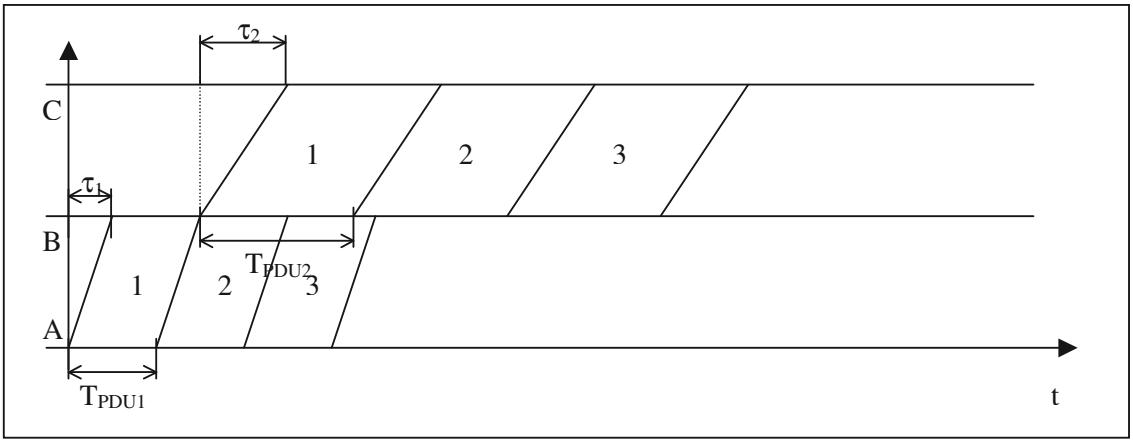
Mettendosi nelle ipotesi del punto 2, calcolare la capacità del sistema ($C_{SISTEMA}$) nel caso in cui $\tau_2 = 100$ *ms* anziché $\tau_2 = 200$ *ms*.

Soluzione

$C_1 = 2000$ byte/sec $\tau_1 = 0{,}075$ sec $\tau_2 = 0{,}2$ sec

1° punto

Suppongo che $C_2 < C_1$ (quindi $T_{PDU2} > T_{PDU1}$), poiché $C_{SISTEMA}$ dipende dalla capacità del link più lento, si avrà:



$$C_{sistema} = \frac{Payload_{Tr}}{T_{PDU2}}$$

sapendo che:

$$T_{PDU2} = \frac{H_{DLC2} + P_{DLC2}}{C_2}$$

si ottiene:

$$C_{sistema} = \frac{Payload_{Tr}}{T_{PDU2}} = \frac{Payload_{Tr}}{H_{DLC2} + P_{DLC2}} \cdot C_2 = 350 \text{ byte/sec}$$

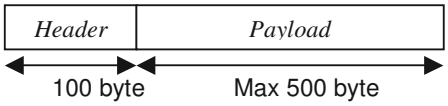
$$C_2 = \frac{H_{DLC2} + P_{DLC2}}{Payload_{Tr}} \cdot C_{sistema} = \frac{100 + 900}{700} \cdot 350 = 500 \text{ byte/sec}$$

tale valore è accettabile in quanto rispetta l'ipotesi: $C_2 < C_1$

2° punto

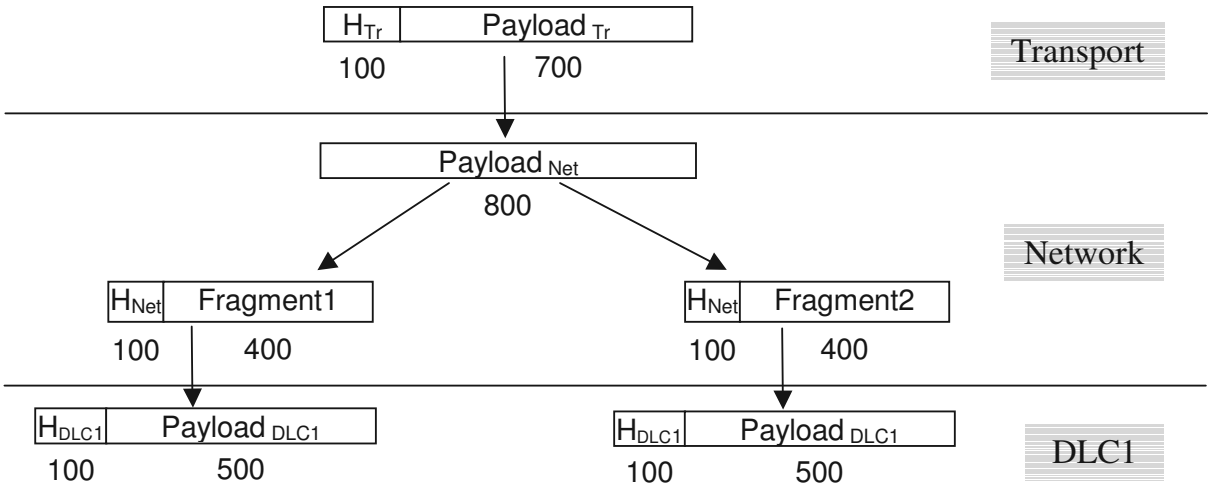
Il livello **DLC1** utilizza un protocollo **non** confermato

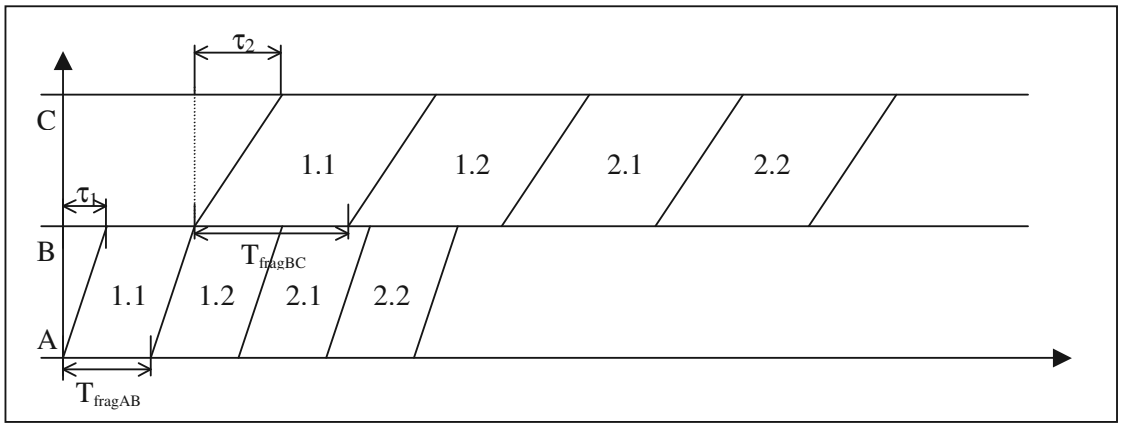
PDU-DATI:



Il livello Network esegue la frammentazione sul nodo mittente, sul nodo destinatario il livello Network corrispondente compierà l'operazione opposta di deframmentazione.

Qui di seguito è illustrata la modalità con cui il livello Network del nodo A esegue l'operazione di frammentazione:





Il tempo che un frammento impiega per andare dal nodo mittente A al nodo intermedio B è calcolato secondo la formula:

$$T_{fragAB} = \frac{H_{DLC1} + P_{DLC1}}{C_1} = \frac{100 + 500}{2000} = 0{,}3\,\text{sec}$$

similmente:

$$T_{fragBC} = \frac{H_{DLC2} + P_{DLC2}}{C_2} = \frac{100+500}{500} = 1{,}2\,\text{sec}$$

Siccome $T_{fragBC} > T_{fragAB}$ è il canale 2 che condiziona il ritmo di arrivo delle PDU, pertanto calcolo $C_{sistema}$ come:

$$C_{sistema} = \frac{Payload_{Tr}}{T_{PDU\,2}} = \frac{Payload_{Tr}}{2 \cdot T_{fragBC}} = \frac{700}{2 \cdot 1{,}2} = 291{,}67 \text{ byte/sec}$$

3° punto

Poiché tutti i protocolli utilizzati dal sistema in esame non prevedono traffico di feedback (sono protocolli di comunicazione non confermati), la capacità del sistema sperimentata al di sopra del livello Transport risulta indipendente dai ritardi dei canali di trasmissione.
Pertanto una variazione del ritardo τ_2 sul canale che collega il nodo B al nodo C non comporta una variazione della $C_{SISTEMA}$ rispetto al risultato ottenuto al punto precedente.

CAPITOLO 2: ACCESSO MULTIPLO A CANALI CONDIVISI

Esercizio 1 *(Appello del 27/02/2004)*

Si supponga di avere un canale ad accesso multiplo con topologia a bus di lunghezza pari a *1 km* ed utilizzante protocollo d'accesso CSMA/CD.
Sapendo che il ritardo di propagazione dei bit sul canale per unità di lunghezza è di *5 ns/m* e che le stazioni connesse su di esso trasmettono messaggi di lunghezza pari a 100 byte, determinare se esistono dei valori del bit-rate *C* del canale tali per cui una collisione potrebbe non essere rilevata.

— · — · —

Soluzione

Innanzitutto devo trovare il τ totale del canale lungo 1 km:

$$\tau = 5ns/m = 5 \cdot 10^{-9} s/m = 5 \cdot 10^{-6} s/km$$

$$\tau_{1Km} = 5 \cdot 10^{-6} s/km \cdot 1km = 5 \cdot 10^{-6} s$$

Determino quindi il bit-rate del canale tale da garantire un tempo minimo di trasmissione dei messaggi da 100 byte pari al tempo di vulnerabilità del protocollo CSMA/CD (ovvero 2τ):

$$2\tau = 2 \cdot 5 \cdot 10^{-6} s = 10^{-5} s$$

$$\frac{100byte}{C} \geq 10^{-5} s$$

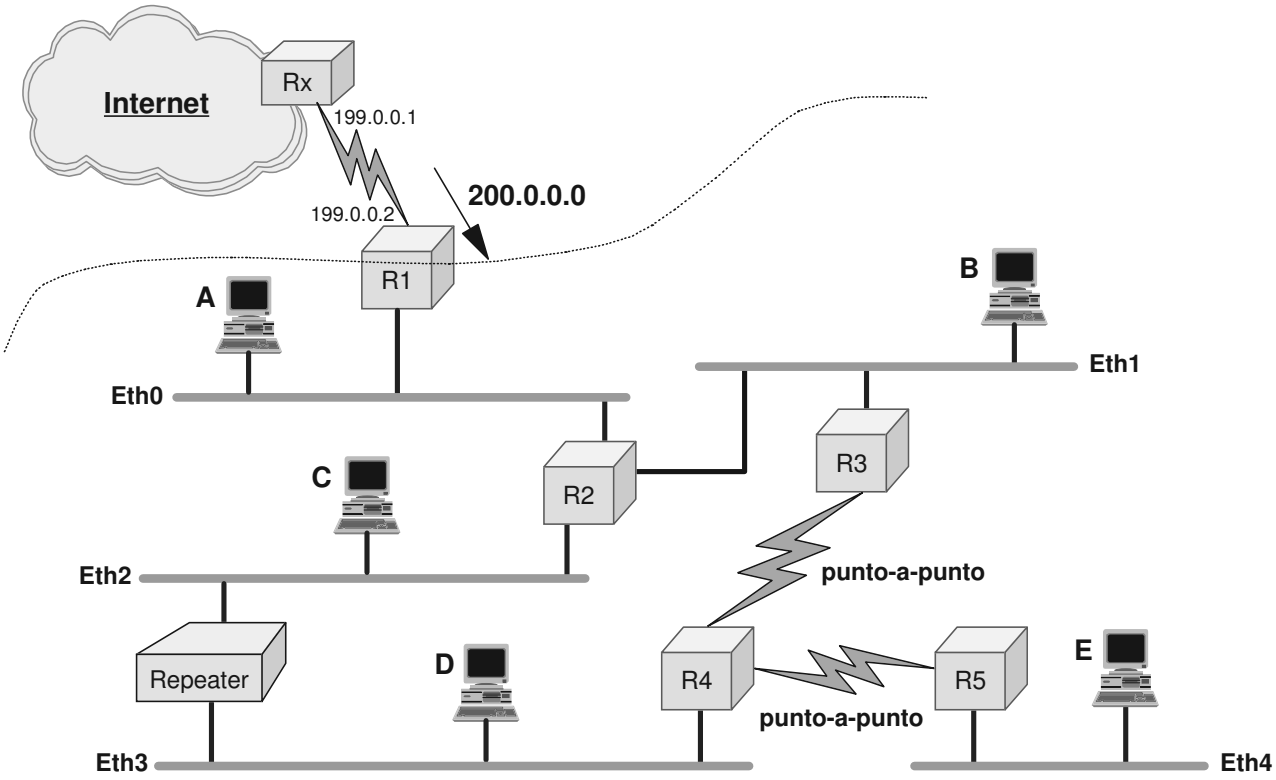
Risolvendo, si trova che la collisione non potrebbe essere rilevata se:

$$C > 10^{-7} byte/s = 10Mbyte/s$$

CAPITOLO 3:
INDIRIZZAMENTO IP
(SUBNETTING)

Esercizio 1 *(Appello del 27/09/2002)*

Sia data la rete riportata sotto. Internet assegna lo spazio di indirizzamento 200.0.0.0 (cioè un Net_Addr di classe C).
Stendere un piano di indirizzamento per la rete indicata nella figura utilizzando tutto il Net_Addr che è stato assegnato da Internet e sapendo che su alcuni link esistono i seguenti vincoli sul numero di host collegabili:
Eth1: almeno 50 host (compreso B)
Eth2: almeno 15 host (compreso C)
Eth3: almeno 30 host (compreso D)
Costruire infine tutte le tabelle di instradamento necessarie.

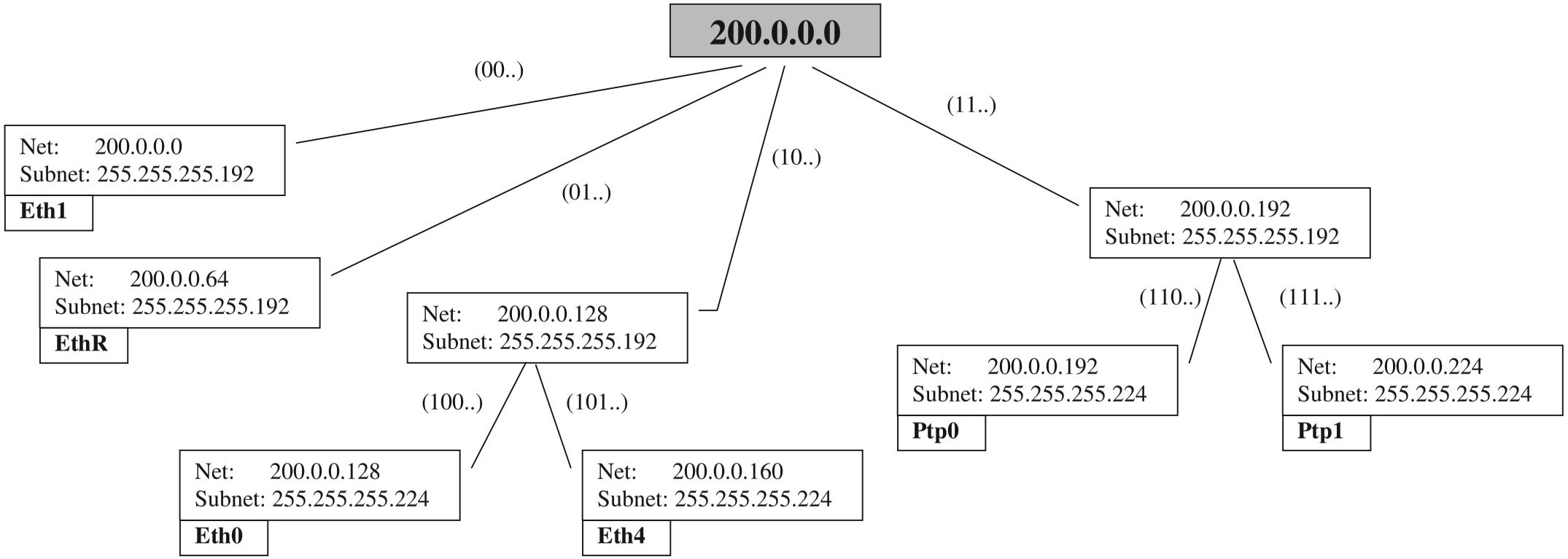


Soluzione

La rete proposta è composta presenta **6 link,** 4 reti Ethernet (Eth0, Eth1, EthR, Eth4) e due punto a punto (Ptp0 e Ptp1). Nella rete è inoltre presente un repeater che collega le ethernet Eth2 e Eth3; poiché quest'ultimo opera a livello Fisico e risulta completamente trasparente ai livelli superiori, le 2 ethernet diventano a tutti gli effetti una unica rete IP (EthR).

Allo scopo di rispettare i vincoli imposti, ad EthR verranno assegnati almeno 45 indirizzi mentre ad Eth1 almeno 50.

L'indirizzo assegnato è di classe C e verrà suddiviso nel seguente schema:



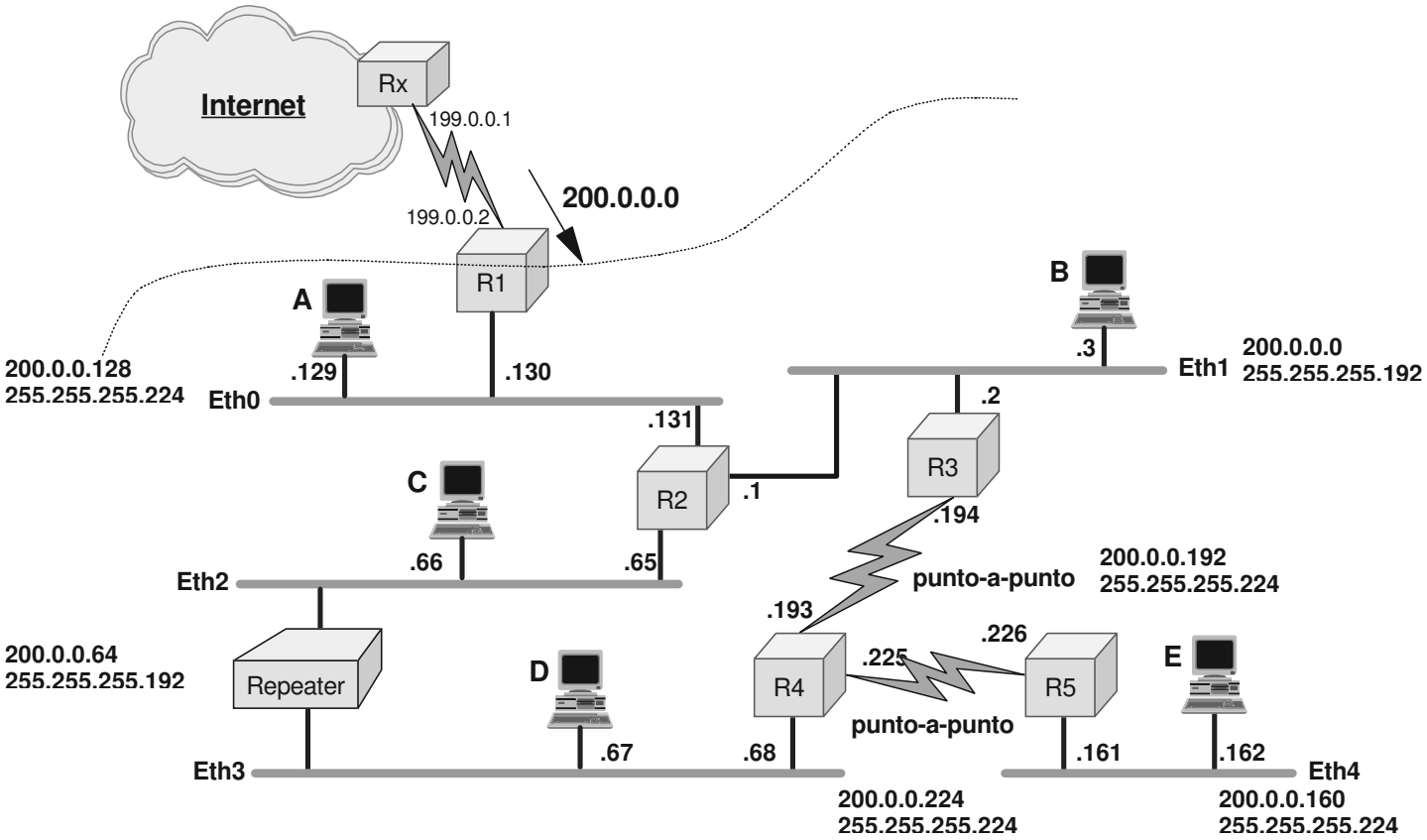


TABELLE DI ROUTING

	Destination	Mask	Next hop
R1	200.0.0.160	255.255.255.224	200.0.0.131
	200.0.0.224	255.255.255.224	200.0.0.131
	200.0.0.192	255.255.255.224	200.0.0.131
	200.0.0.64	255.255.255.192	200.0.0.131
	200.0.0.0	255.255.255.192	200.0.0.131
	Default	----------	199.0.0.1

	Destination	Mask	Next hop
A	200.0.0.160	255.255.255.224	200.0.0.131
	200.0.0.224	255.255.255.224	200.0.0.131
	200.0.0.192	255.255.255.224	200.0.0.131
	200.0.0.64	255.255.255.192	200.0.0.131
	200.0.0.0	255.255.255.192	200.0.0.131
	Default	----------	200.0.0.130

	Destination	Mask	Next hop
R2	200.0.0.160	255.255.255.224	200.0.0.68
	200.0.0.224	255.255.255.224	200.0.0.68
	200.0.0.192	255.255.255.224	200.0.0.68
	Default	----------	200.0.0.130

	Destination	Mask	Next hop
B	200.0.0.160	255.255.255.224	200.0.0.2
	200.0.0.224	255.255.255.224	200.0.0.2
	200.0.0.192	255.255.255.224	200.0.0.2
	Default	----------	200.0.0.1

	Destination	Mask	Next hop
C,D	200.0.0.160	255.255.255.224	200.0.0.68
	200.0.0.224	255.255.255.224	200.0.0.68
	200.0.0.192	255.255.255.224	200.0.0.68
	Default	----------	200.0.0.65

	Destination	Mask	Next hop
R3	200.0.0.160	255.255.255.224	200.0.0.193
	200.0.0.224	255.255.255.224	200.0.0.193
	Default	----------	200.0.0.1

	Destination	Mask	Next hop
R4	200.0.0.160	255.255.255.224	200.0.0.226
	Default	----------	200.0.0.65

	Destination	Mask	Next hop
R5	Default	----------	200.0.0.225

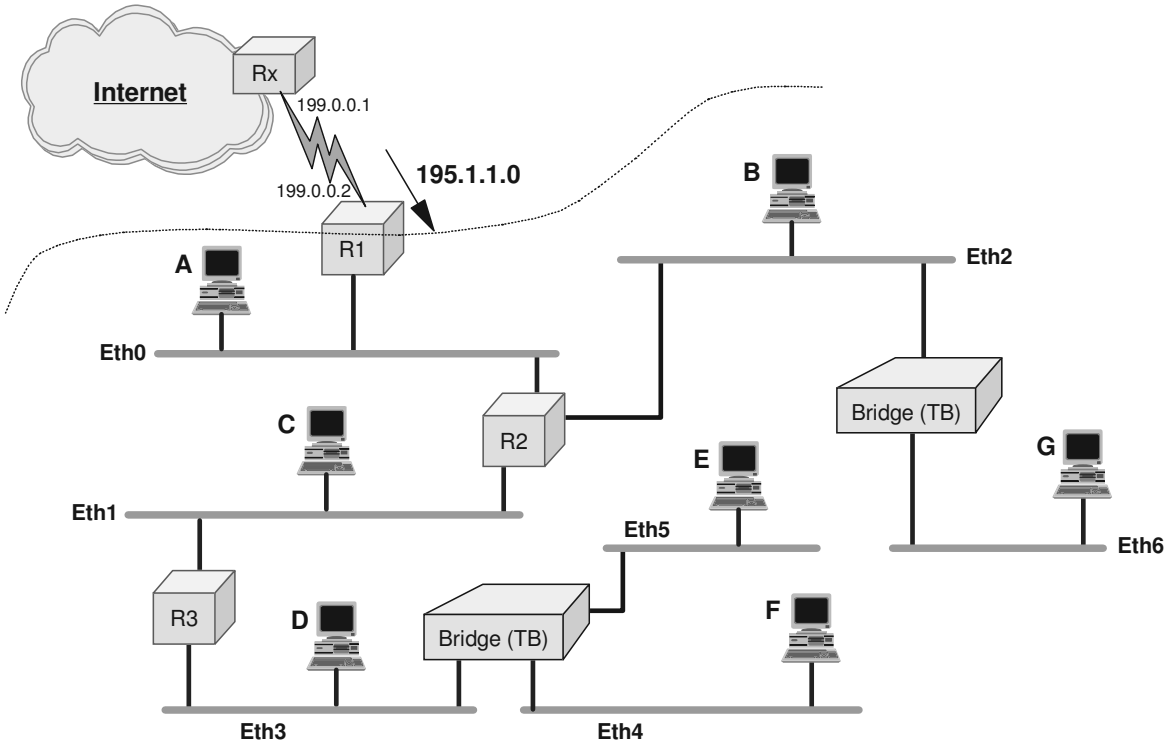
	Destination	Mask	Next hop
E	Default	----------	200.0.0.161

Esercizio 2 *(Appello del 10/02/2004)*

Sia data la rete IPv4 indicata nella figura qui di seguito. Su alcuni segmenti LAN esistono dei vincoli circa il numero di host che devono poter essere collegati:
Eth3: n.28 host (compreso D)
Eth4: n.28 host (compreso F)
Eth5 n.55 host (compreso E)
Eth6 n.55 host (compreso G)

Internet assegna lo spazio di indirizzamento IPv4 195.1.1.0 (Classe C). Stendere un piano di indirizzamento per la rete indicata nella figura (illustrando chiaramente i criteri utilizzati, nonché i singoli valori delle subnet_mask) coerentemente con lo spazio che è stato assegnato e i vincoli indicati, costruendo altresì tutte le tabelle di instradamento IPv4 necessarie.

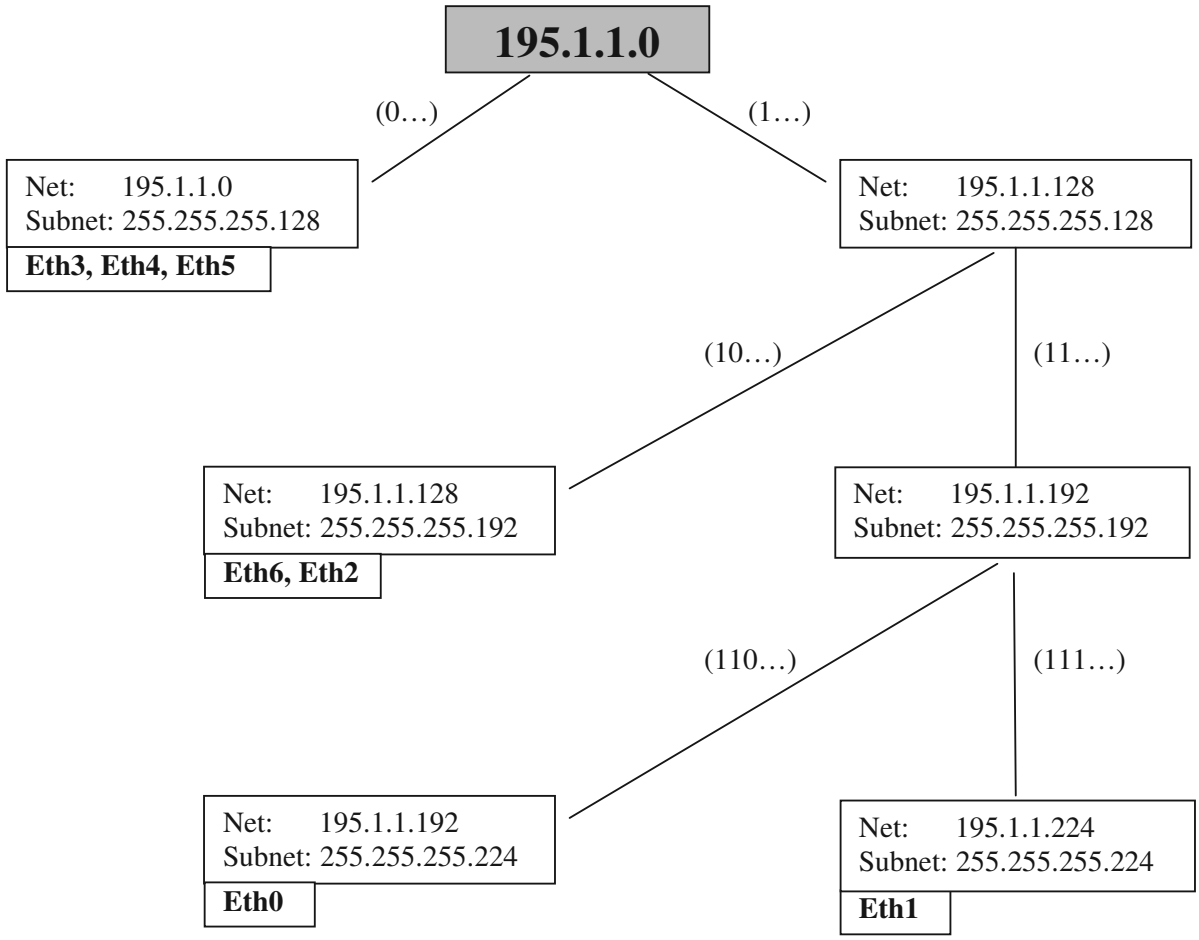
NOTA PER LO SVOLGIMENTO DELL'ESERCIZIO: Gli indirizzi dei vari nodi possono essere riportati direttamente sullo schema qui sotto.



Soluzione

La rete proposta è composta da 7 link; tuttavia alcuni di questi sono connessi tra loro tramite dei Trasparent Bridge, ovvero dei dispositivi che operano a livello MAC in maniera trasparente ai livelli superiori. Pertanto le Ethernet 3, 4 e 5 costituiranno un'unica rete e lo stesso dicasi per Eth2 e Eth6.
Per rispettare i vincoli imposti Eth3,4,5 dovrà disporre di almeno 111 indirizzi ed Eth2,6 di almeno 55; non sono previsti vincoli invece per il numero di host minimi per Eth0 ed Eth1

L' indirizzo assegnato è di classe C e verrà suddiviso secondo il seguente schema:



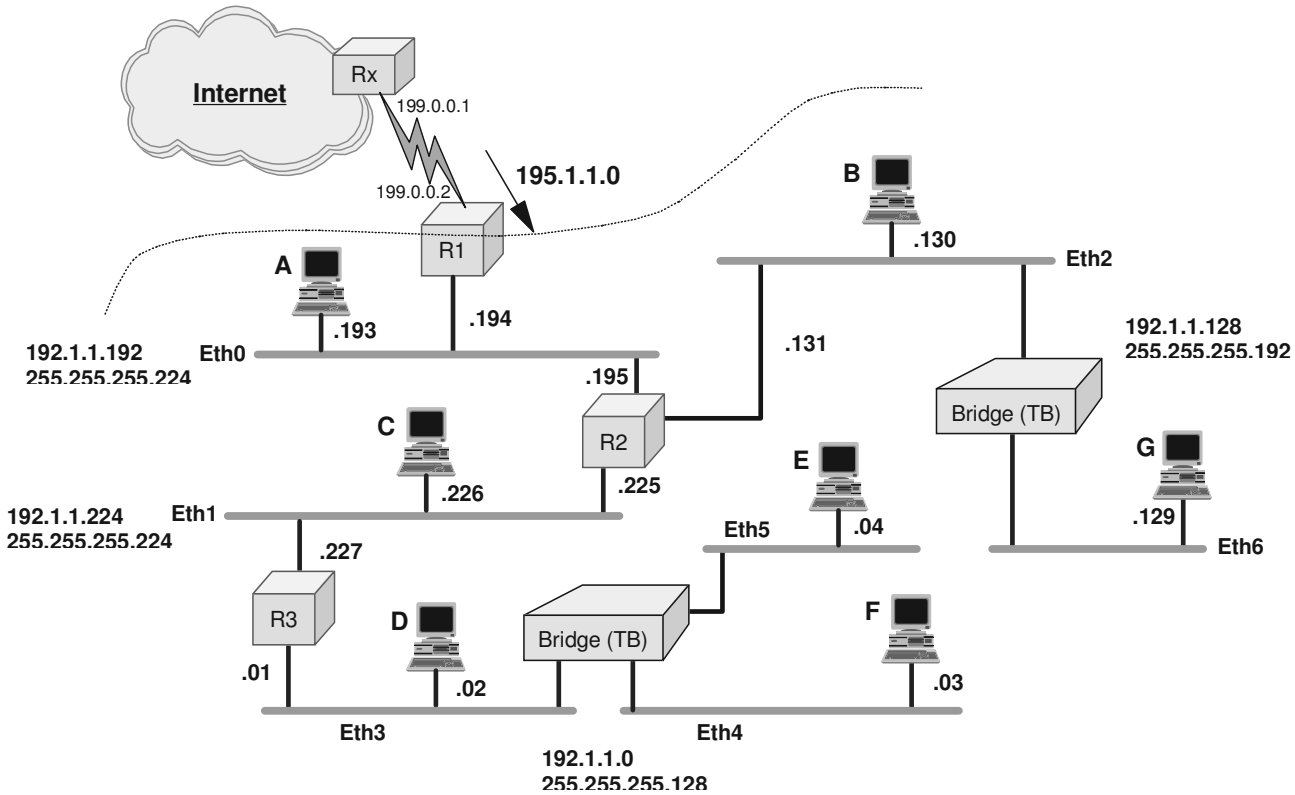


TABELLE DI ROUTING

	Destination	Mask	Next hop
R1	195.1.1.0 (Eth3,4,5)	255.255.255.128	195.1.1.195
	195.1.1.128 (Eth2,6)	255.255.255.192	195.1.1.195
	195.1.1.224 (Eth1)	255.255.255.224	195.1.1.195
	Default	----------	199.0.0.1

	Destination	Mask	Next hop
R2	195.1.1.0 (Eth3,4,5)	255.255.255.128	195.1.1.227
	Default	----------	195.1.1.194

	Destination	Mask	Next hop
R3	Default	----------	195.1.1.225

	Destination	Mask	Next hop
Host A	195.1.1.0 (Eth3,4,5)	255.255.255.128	195.1.1.195
	195.1.1.128 (Eth2,6)	255.255.255.192	195.1.1.195
	195.1.1.224 (Eth1)	255.255.255.224	195.1.1.195
	Default	----------	195.1.1.194

	Destination	Mask	Next hop
Host B, G	Default	----------	195.1.1.131

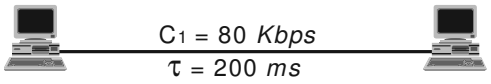
	Destination	Mask	Next hop
Host C	195.1.1.0 (Eth3,4,5)	255.255.255.128	195.1.1.227
	Default	----------	195.1.1.225

	Destination	Mask	Next hop
Host D, E, F	Default	----------	195.1.1.1

CAPITOLO 4: ANALISI DELLE PRESTAZIONI DELLE RETI TCP/IP

Esercizio 1 *(Appello del 27/09/2002)*

Si supponga di avere 2 host TCP/IP connessi da una linea full-duplex punto-a-punto con capacità C = 80 Kbps e ritardo τ = 200ms, i quali comunicano utilizzando il protocollo di trasport UDP.



Le caratteristiche dei protocolli sono le seguenti:
Lunghezza_Max_Payload$_{UDP}$ = 2000 byte
H_{UDP} = 8 Byte
H_{IP} = 20 Byte
H_{DLC} = 18 Byte

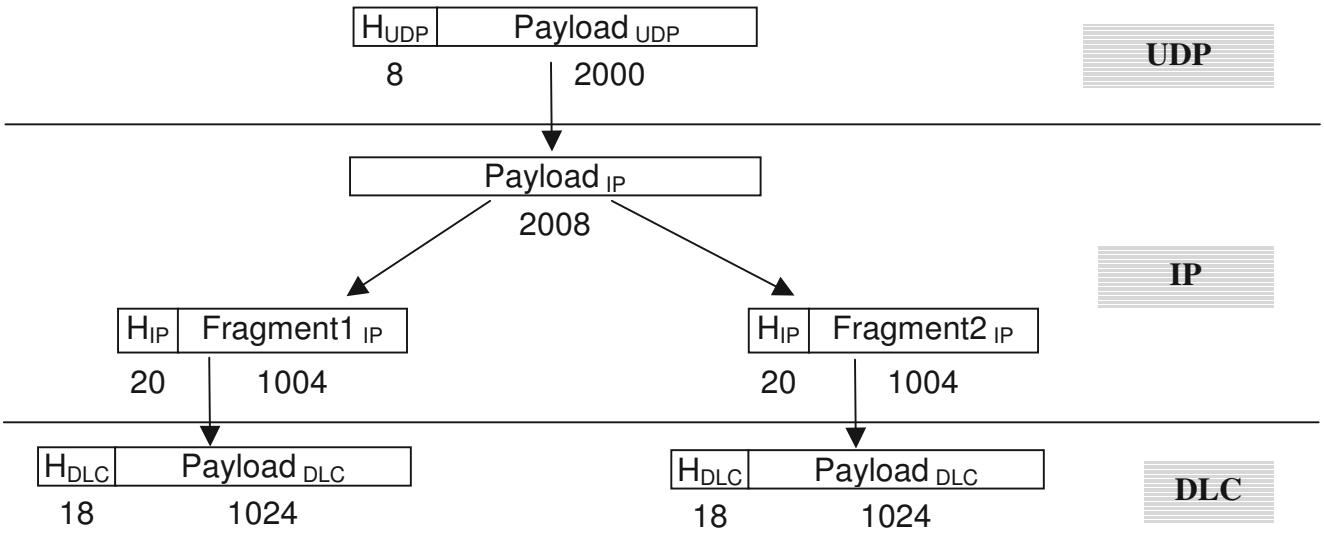
Domande:

Supponendo di frammentare i *datagram* IP in *n fragment* di ugual dimensione, determinare $C_{SISTEMA}$ osservata al di sopra di UDP nei casi in cui *n = 2, 3, 4*, determinando altresì le MTU da imporre sul DLC affichè avvengano tali frammentazioni.

Soluzione

n = 2:

L'IP deve eseguire la frammentazione nei confronti del livello sottostante secondo il seguente schema:



Come si può osservate il valore dell'MTU è dato da:

$$MTU = \frac{Payload_{IP}}{n} + H_{IP} = \frac{2008}{2} + 20 = 1024 Byte$$

La capacità del sistema è data dalla seguente formula:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}}$$

Il tempo di trasmissione è dato dal tempo di trasferimento di due frammenti:

$$T_{TRASM} = 2 \cdot T_{FRAM}$$

Il tempo di trasferimento di un frammento è dato da:

$$T_{FRAM} = \frac{MTU_{DLC} + H_{DLC}}{C} = \frac{1024 + 18}{10000} = 0{,}104 s$$

Quindi:

$$T_{TRASM} = 2 \cdot T_{FRAM} = 2 \cdot 0{,}104 = 0{,}208 s$$

E la capacità del sistema risulta essere:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}} = \frac{2000}{0{,}208} = 9615 \, Byte/s$$

n = 3:

Il livello IP deve eseguire la frammentazione nei confronti del DLC.
Vengono costruiti tre frammenti IP, due con payload di 670 Byte e uno di 668 Byte. Tutti e tre hanno 20 Byte di header. La MTU è data quindi da 690 Byte (in realtà i tre frammenti non sono tutti uguali, ma l'errore commesso "aggiungendo" 2 Byte all'ultimo è trascurabile).

$$MTU = \frac{Payload_{IP}}{n} + H_{IP} = \frac{2008}{3} + 20 = 690 Byte$$

La capacità del sistema è data dalla seguente formula:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}}$$

Il tempo di trasmissione è dato dal tempo di trasferimento di tre frammenti:

$$T_{TRASM} = 3 \cdot T_{FRAM}$$

Il tempo di trasferimento di un frammento è dato da:

$$T_{FRAM} = \frac{MTU_{DLC} + H_{DLC}}{C} = \frac{690+18}{10000} = 0{,}071s$$

Quindi:

$$T_{TRASM} = 3 \cdot T_{FRAM} = 3 \cdot 0{,}071 = 0{,}213s$$

E la capacità del sistema risulta essere:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}} = \frac{2000}{0{,}213} = 9390 \, Byte/s$$

n = 4:

In questo caso vengono costruiti 4 frammenti a livello IP con payload di 502 Byte e 20 Byte di header.
L'MTU, ovvero la payload del DLC, risulta quindi di 522 Byte.

$$MTU = \frac{Payload_{IP}}{n} + H_{IP} = \frac{2008}{4} + 20 = 522Byte$$

La capacità del sistema è data dalla seguente formula:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}}$$

Il tempo di trasmissione è dato dal tempo di trasferimento di quattro frammenti:

$$T_{TRASM} = 4 \cdot T_{FRAM}$$

Il tempo di trasferimento di un frammento è dato da:

$$T_{FRAM} = \frac{MTU_{DLC} + H_{DLC}}{C} = \frac{522+18}{10000} = 0{,}054s$$

Quindi:

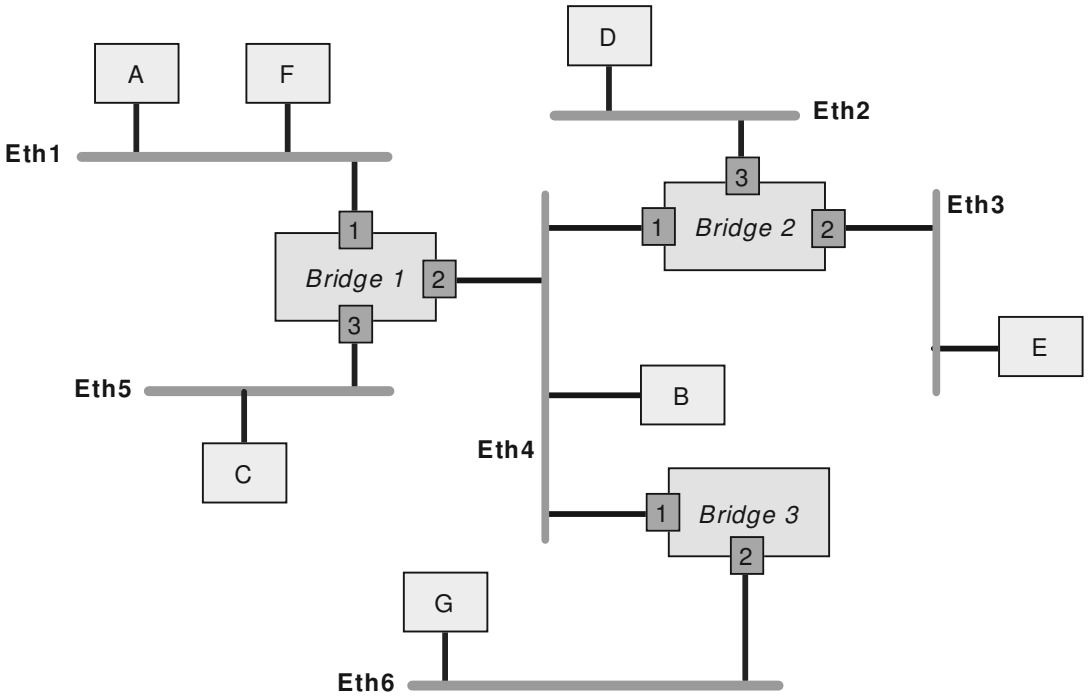
$$T_{TRASM} = 4 \cdot T_{FRAM} = 4 \cdot 0{,}054 = 0{,}216s$$

E la capacità del sistema risulta essere:

$$C_{sistema} = \frac{L_{UDP}}{T_{TRASM}} = \frac{2000}{0{,}216} = 9260 \; Byte/s$$

CAPITOLO 5: DISPOSITIVI DI INTERCONNESSIONE

Esercizio 1



Sia data la rete indicata sopra, formata dall'interconnessione di 6 segmenti LAN Ethernet. I dispositivi di interconnessione sono i Bridge Transparent *Bridge1*, *Bridge2*, *Bridge3*.

Si ipotizza che la rete sia appena stata avviata e che nessuna frame sia ancora stata trasmessa.

Spiegare come avviene la propagazione delle frame sulla rete, evidenziando inoltre il contenuto delle tabelle sui nodi (*station cache*), quando avvengono in sequenza le seguenti quattro trasmissioni:
1^ trasmissione: da A a G ;
2^ trasmissione: da A a B ;
3^ trasmissione: da B a A ;
4^ trasmissione: da G a B ;

Soluzione

Prima di analizzare le modalità di propagazione delle frame, è opportuno ricordare la modalità di funzionamento di un Trasparent Bridge. Questo dispositivo, che opera a livello MAC, riceve ogni pacchetto trasmesso sui segmenti e associa il MAC_Addr del mittente alla porta sulla quale è stato ricevuto il messaggio in un apposita tabella. A questo punto, se il MAC_Addr del destinatario è presente anch'esso nella station cache, il Bridge provvede a ritrasmettere il pacchetto sulla porta ad esso associata (a patto ovviamente che questa sia diversa dalla paorta dalla quale è stato ricevuto). Se invece l'indirizzo MAC del destinatario non viene trovato (ovvero il bridge non ha di recente ricevuto pacchetti da quest'ultimo), il pacchetto viene ritrasmesso su tutte le porte eccetto quella da cui è stato ricevuto.

1^ Trasmissione (A → G)

La frame viene inviata da A sul link Eth0 e ricevuta quindi dal Bridge1 sulla porta 1; questi tuttavia è stato appena acceso e pertanto non contiene nella station cache alcuna informazione circa la posizione di G. La frame viene quindi ritrasmessa sulle porte 2 (Eth4) e 3 (Eth5) ed è quindi ricevuta anche da Bridge1 e Bridge2 (entrambi sulla porta 1). Anche questi due dispositivi non dispongono di alcuna informazione su G e si limitano a loro volta a ritrasmettere la frame sulle altre porte (ovvero su Eth2, Eth3 ed Eth6, dove è connesso G).

	Host	Port
Bridge1	A	1

	Host	Port
Bridge2	A	1

	Host	Port
Bridge3	A	1

2^ Trasmissione (A → B)

La frame è inviata da A sul link Eth0 e ricevuta quindi dal Bridge1 sulla porta 1; la riga nella station cache che associa A alla porta 1 è già presente e quindi semplicemente viene fatto ripartire l'ageing time. Non essendo però nota la porta sulla quale si trova B, la frame viene ritrasmessa sia sulla porta 2 che sulla porta 3. Il pacchetto a questo punto è giunto anche su Eth4 e quindi viene ricevuto correttamente sia dal destinatario (B) che dai Bridge 2 e 3. I due dispositivi potrebbero limitarsi a scartare la frame ma nelle loro station cache non è presente una entry che associ B alla porta 1; anche'essi pertanto aggiornano l'ageing time della riga di A e, come accaduto nella prima trasmissione, ritrasmettono il pacchetto su tutte le altre porte.

	Host	Port
Bridge1	A	1

	Host	Port
Bridge2	A	1

	Host	Port
Bridge3	A	1

3^ Trasmissione (B → A)

B trasmette la frame su Eth4 e questa viene ricevuta da tutte e tre i Bridge. Bridge2 e Bridge3 aggiungono pertanto alla station cache le informazioni riguardanti B per poi scartare però il pacchetto: infatti la porta associata ad A è la medesima di B. Bridge1 invece inserirà una nuova riga per B nella station cache e quindi inoltrerà lungo la porta 1 la frame destinata ad A, come indicato nella tabella.

	Host	Port
Bridge1	A	1
	B	2

	Host	Port
Bridge2	A	1
	B	1

	Host	Port
Bridge3	A	1
	B	1

4^ Trasmissione (G → B)

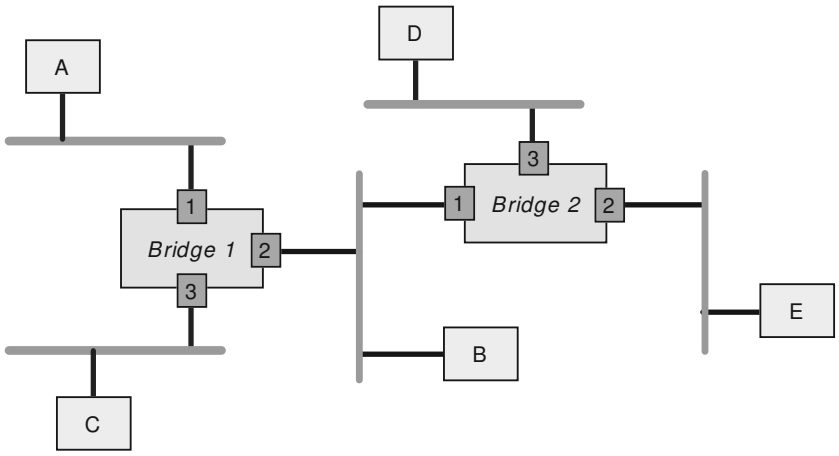
La frame trasmessa da G lungo Eth6 è ricevuta da Bridge3 sulla porta 2 e ritrasmessa, come indicato nella station cache, sulla porta 1. A questo punto il pacchetto è ricevuto dal destinatario B e dai TB Bridge1 e Bridge2: tuttavia, per entrambi, nelle tabelle della station cache le porte associate a B coincidono con quella dalla quale sono giunti i dati e pertanto la frame viene eliminata dopo che sono state aggiunte le righe che associano G alla relativa porta.

	Host	Port
Bridge1	A	1
	B	2
	G	2

	Host	Port
Bridge2	A	1
	B	1
	G	1

	Host	Port
Bridge3	A	1
	B	1
	G	2

Esercizio 2 *(Appello del 25/02/2003)*



Sia data la rete indicata qui sotto, formata dall'interconnessione di 5 segmenti LAN Ethernet. I dispositivi di interconnessione sono i Bridge Transparent *Bridge1* e *Bridge2*.
La rete è stata appena avviata, quindi nessuna frame è ancora stata trasmessa. Spiegare come avviene la propagazione delle frame sulla rete, evidenziando inoltre il contenuto delle tabelle sui nodi (*station cache*), quando avvengono le seguenti trasmissioni:
1^ trasmissione: da A a D ;
2^ trasmissione: da E a A ;
3^ trasmissione: da E a C ;
4^ trasmissione: da B a D ;

— · — · —

Soluzione

I Bridge ascoltano quello che transita sulle loro porte e memorizzano nella *station cache* la posizione del mittente.
In fase di inoltro i Trasparent Bridge consultano le loro tabelle per sapere se conoscono a che porta corrisponde il destinatario. Se non hanno questa informazione inoltrano il pacchetto verso tutte le porte, esclusa quella di provenienza.

1^ Trasmissione (A → D)

Tutti e 2 i Bridge imparano la posizione del nodo A (il mittente) e in tutte le station cache si aggiunge una riga che definisce la posizione di A. In fase di inoltro, nessun Bridge conosce la posizione di D e quindi tutti inoltrano a tutte le porte esclusa quella di provenienza.

	Host	**Port**
Bridge1	A	1

	Host	**Port**
Bridge2	A	1

2^ Trasmissione (E → A)

Tutti e 2 i Bridge imparano la posizione del nodo E (il mittente) e in tutte le station cache si aggiunge una riga che definisce la posizione di E. In fase di inoltro, i Bridge conoscono la posizione di A e quindi inoltrano alle porta indicata nelle loro Station Cache.

	Host	Port
Bridge1	A	1
	E	2

	Host	Port
Bridge2	A	1
	E	2

3^ Trasmissione (E → C)

Tutti e 2 i Bridge conoscono la posizione del nodo E (mittente), per cui non aggiungono nessuna nuova riga nelle loro Station Cache. In fase di inoltro, nessun Bridge conosce la posizione di C e quindi tutti inoltrano a tutte le porte esclusa quella di provenienza.

	Host	Port
Bridge1	A	1
	E	2

	Host	Port
Bridge2	A	1
	E	2

4^ Trasmissione (B → D)

Tutti e 2 i Bridge imparano la posizione del nodo B (il mittente) e in tutte le station cache si aggiunge una riga che definisce la posizione di B. In fase di inoltro, nessun Bridge conosce la posizione di D e quindi tutti inoltrano a tutte le porte esclusa quella di provenienza.

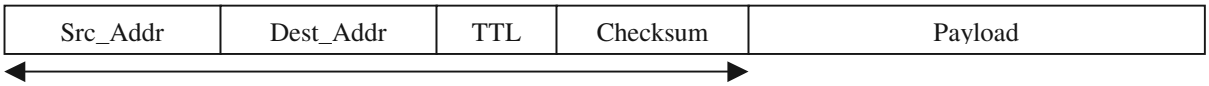
	Host	Port
Bridge1	A	1
	E	2
	B	2

	Host	Port
Bridge2	A	1
	E	2
	B	1

APPENDICE: DOMANDE DI TEORIA

Esercizio 1 *(Appello del 08/09/2003)*

Sia dato un protocollo di comunicazione di livello 3, il quale prevede un unico formato di PDU (PDU-DATI) riportato qui sotto (la doppia freccia evidenzia la parte *header*).



Significato e lunghezze dei campi della PDU-DATI del protocollo:

Src_Addr (16 bit) = Indirizzo entità mittente
Dest_Addr (16 bit) = Indirizzo entità destinataria
TTL (4 bit) = Viene impostato dal mittente ad un valore >0 e decrementato di 1 ad ogni attraversamento di un nodo intermedio. Se TTL=0 su un nodo intermedio allora il pacchetto viene eliminato.
Checksum (8 bit) = Campo per il controllo dell'integrità della sola porzione *header*
Payload = PDU del livello superiore

Domande

1. Dire quali di queste funzioni (motivando sempre le risposte) possono essere svolte da tale protocollo:
   a. Controllo degli errori (*Error Control*): se necessario fare i vari sottocasi
   b. Frammentazione delle PDU (*Fragmentation*)
   c. Controllo di Flusso (*Flow Control*)
   d. Consegna in sequenza (*Ordered Delivery*)
   e. Multiplazione dei traffici originati da entità diverse di livello 4 (*Protocol Multiplexing*)

2. Supponendo di voler utilizzare questo protocollo per costruire una rete di grandi dimensioni, esistono delle limitazioni circa il numero di nodi configurabili e/o la topologia della rete ?

Costruendo una rete complessa con un simile protocollo di livello 3 é possibile utilizzare indifferentemente una tecnica di forwarding di tipo *Source-Route-Forwarding* o *Destination-Based-Forwarding* ? Spiegare le motivazioni.

Soluzione

Punto 1

a) Può effettuare un controllo di integrità sulla Header grazie al campo Checksum. Non può correggere errori sulla parte Payload, ne eventualmente attuare una ritrasmissione con tecniche ARQ perché mancano i campi destinati a questo scopo.

b) No, mancano i campi per numerare la PDU (necessario per identificare a quale PDU appartiene un certo frammento) e i frammenti stessi (campo necessario per ricomporre i frammenti ricevuti nella sequenza corretta); inoltre non è previsto alcun campo che permetta al destinatario di sapere se il frammento ricevuto era l'ultimo né tantomeno di identificare il pacchetto ricevuto come frammento piuttosto che come PDU completa.

c) No. Come già visto, non è possibile utilizzare le tecniche ARQ per gestire il flusso di dati e non è nemmeno possibile utilizzare un controllo di flusso di tipo Choke Packet: non sono

previsti infatti campi “speciali” utilizzabili per comunicazioni di servizio. In un caso come questo è possibile solamente utilizzare sul mittente tecniche di controllo di flusso che non richiedono feedback (ad esempio Token Bucket).

d) No, manca un campo per la numerazione delle Pdu.

e) No, manca un campo per la scelta del protocollo di Livello 4 opportuno.

Punto 2

Sì, il campo TTL può essere impostato al valore massimo 15 ($=2^4-1$): non si potranno perciò avere percorsi con più di 15 nodi intermedi.
Inoltre la dimensione del campo Address di 16 bit vincola il numero massimo di host contemporaneamente connessi a 65536 ($=2^{16}$)

Punto 3

Posso utilizzare solo la tecnica di tipo Destination-Based-Forwording perché nella Header è possibile indicare solo il destinatario e non tutti i nodi per raggiungerlo, come esigerebbe invece la tecnica Source-Route-Forwording.

Esercizio 2 *(Appello del 30/06/2003)*

Illustrare 2 tecniche di *forwarding* utilizzate nelle reti a commutazione di pacchetto, indicando per ciascuna un esempio di utilizzo in dispositivi reali.

— · — · —

Soluzione

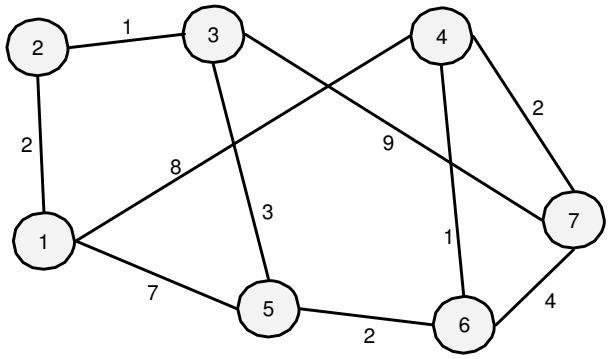
Una volta identificato il percorso che una k-PDU deve seguire per giungere all'entità destinataria, tutti i nodi lungo il cammino (eccetto il destinatario) devono procedere al suo inoltro su quel percorso. Questa operazione prende il nome di forwarding (o commutazione) e può avvenire in modalità differenti, in base al tipo di tecnica utilizzata dal sistema e dai dispositivi.
La tecnica *Source Route Forwarding* prevede ad esempio che l'intero percorso da seguire venga individuato e memorizzato sul nodo mittente e quindi inserito da quest'ultimo in un campo opportuno (`Routing_Info`) della header di livello k. Compito dei nodi intermedi sarà esclusivamente quello di riconoscersi nella `Routing_Info` della k-PDU e spedirla quindi verso il nodo successivo indicato nel percorso. L'utilizzo di questa tecnica presenta tuttavia alcuni inconvenienti: in primo luogo la necessità di specificare tutti i passi all'interno della header limita notevolmente il numero massimo di passi specificabili ed inoltre il carico computazionale sul nodo mittente è notevole, dovendo quest'ultimo ricreare una mappa dell'intera rete ed individuare da solo il cammino completo più breve. La stazione mittente nelle LAN Token Ring può utilizzare ad esempio una forma di routing dinamico centralizzato basato su una ricerca del percorso di tipo flooding (*All-Route Broadcast Route Determination*) per apprendere il percorso che la separa dalla stazione destinataria ed inserirlo nelle frame trasmesse; i Source Route Bridge non dovranno far altro quindi che commutare i dati in base al percorso indicato.
Un'altra tecnica di forwarding, quella *Destination-Based*, prevede invece che in ogni k-PDU sia indicato solamente l'indirizzo globale dell'entità destinataria e che i nodi stessi commutino sulla base di quest'ultimo, individuando ciascuno in maniera autonoma il percorso lungo il quale inoltrare il pacchetto. Affinché questo possa avvenire, su tutti i nodi è richiesta la presenza di una struttura dati organizzata nella forma di database distribuito e contenente informazioni sufficienti ad individuare il passo successivo che i dati dovranno compiere per proseguire nel cammino verso la destinazione. La tecnica di *forwarding Destination-Based* è comune a dispositivi di interconnessione quali i Trasparent Bridge ed i Router. Pur appoggiandosi a tecniche di routing differenti, entrambi i dispositivi dispongono infatti al loro interno di una tabella, organizzata nella forma `Dest_MAC_Addr - Port` (station cache dei bridge) o `Dest_Addr - Next_Hop` (routing table dei router), che permette loro di commutare i pacchetti esclusivamente sulla base dell'indirizzo del destinatario.

PARTE 2: APPROFONDIMENTI DEL CORSO (3 CFU)

CAPITOLO 1:
INSTRADAMENTO CON ALGORITMO DI DIJKSTRA

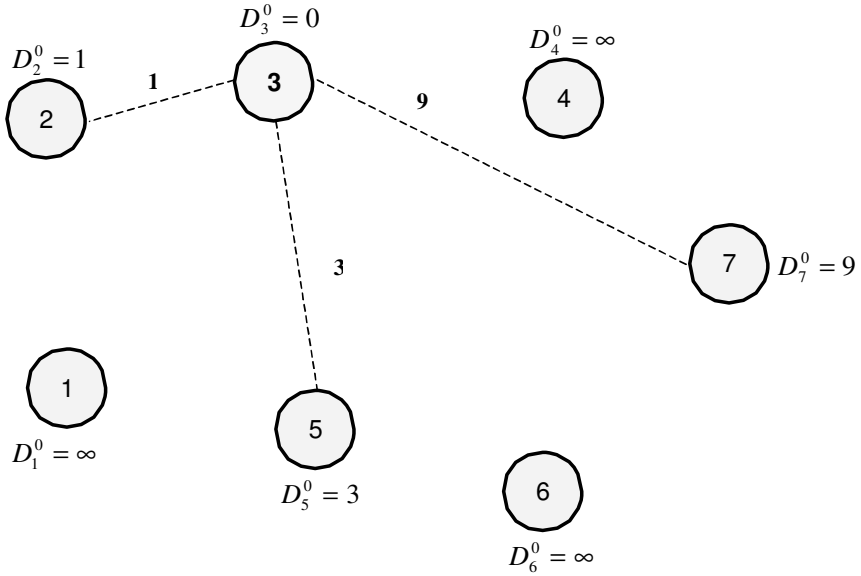
Esercizio 1 *(Appello del 27/09/2002)*

Sia dato il grafo *G=(N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Dijkstra, calcolare il percorso a costo minimo da ogni nodo di *N* al nodo 3 (destinatario). Indicare con rigore i vari passi dell'algoritmo, utilizzando, se possibile, le notazioni usate a lezione.



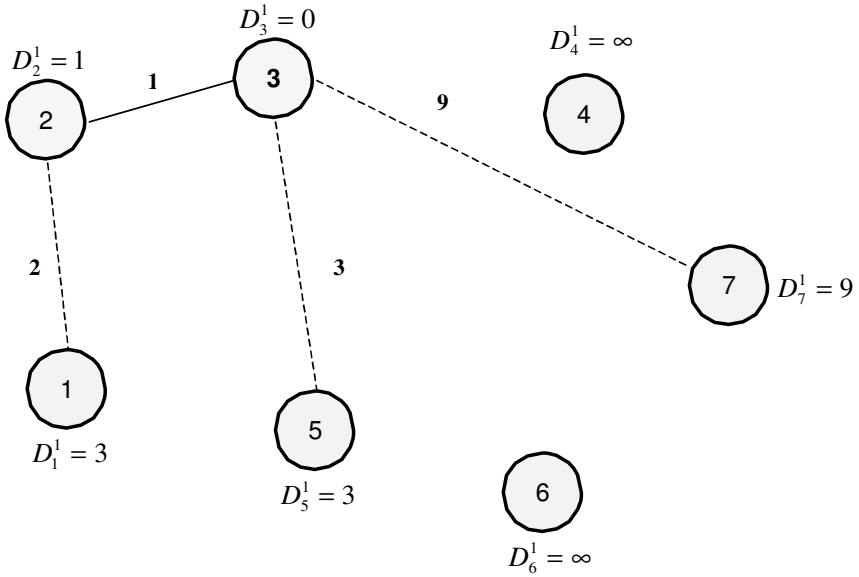
Soluzione

PASSO 0: valuto i nodi 2 – 5 –7 collegati con il nodo 3



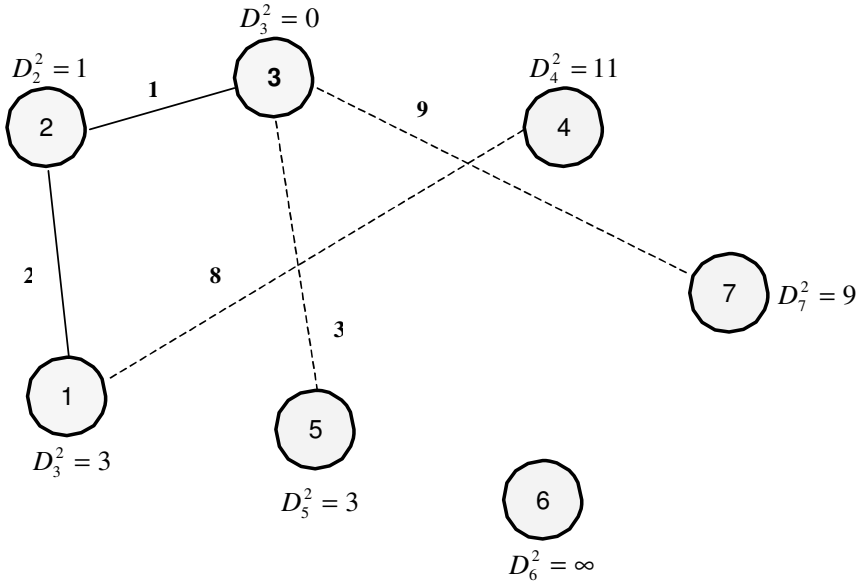
S = {3}

PASSO 1: Scelgo il nodo 2, lo aggiungo all'insieme S e valuto il nodo 1 ad esso collegato



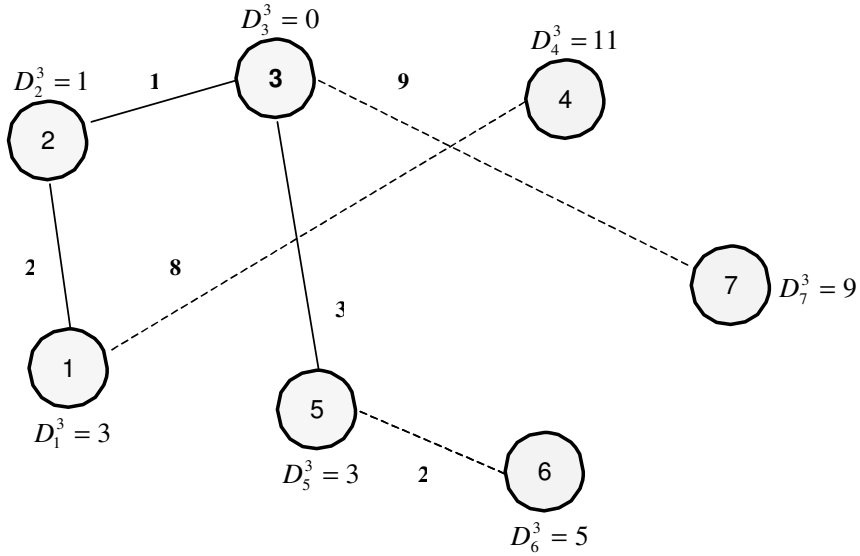
S = {3, 2}

PASSO 2: Scelgo il nodo 1, lo aggiungo all'insieme S e valuto i nodi 4 - 5 ad esso collegati



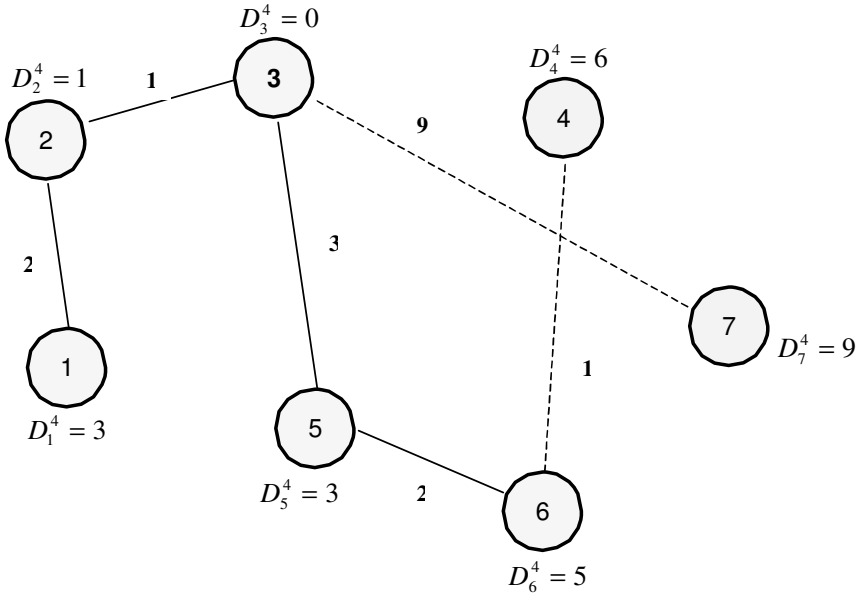
S = {3, 2, 1}

PASSO 3: Scelgo il nodo 5, lo aggiungo all'insieme S e valuto il nodo 6 ad esso collegato



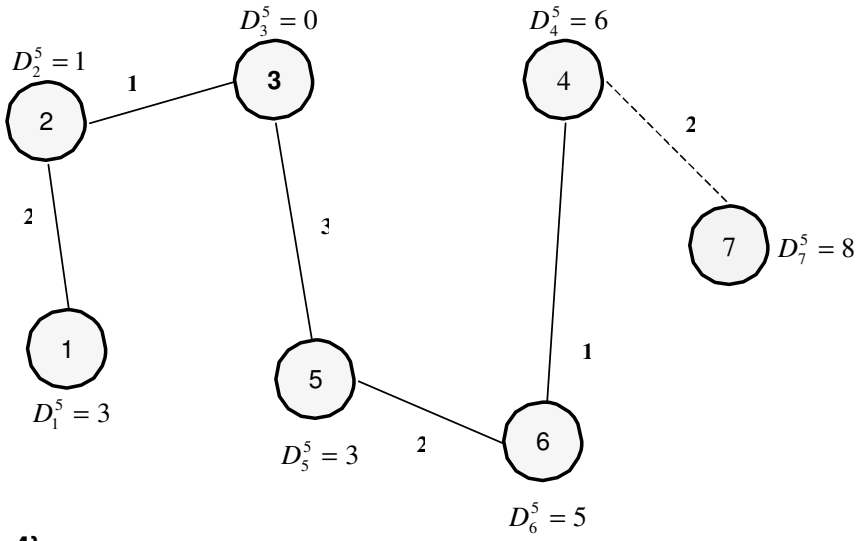
S = {3, 2, 1, 5}

PASSO 4: Scelgo il nodo 6, lo aggiungo all'insieme S e valuto i nodi 4 - 7 ad esso collegati



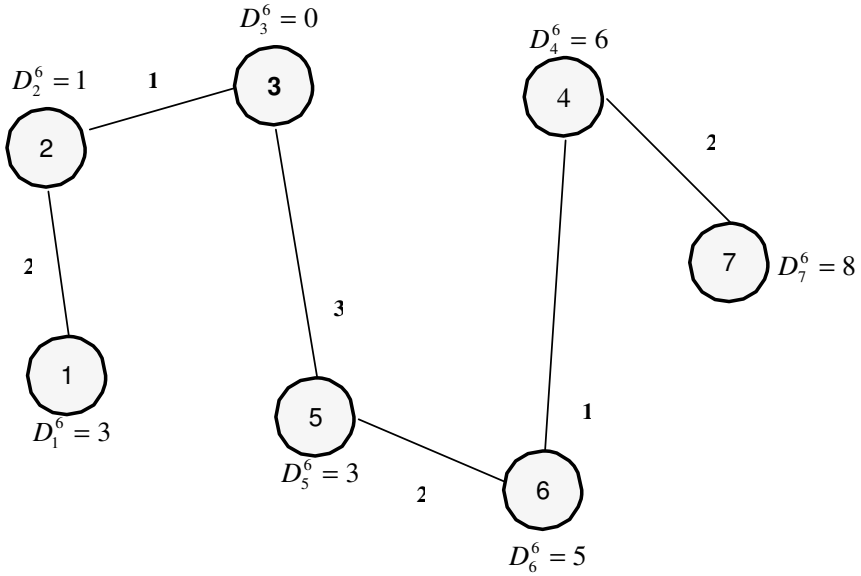
S = {3, 2, 1, 5, 6}

PASSO 5: Scelgo il nodo 4, lo aggiungo all'insieme S e valuto il nodo 7 ad esso collegato



S = {3, 2, 1, 5, 6, 4}

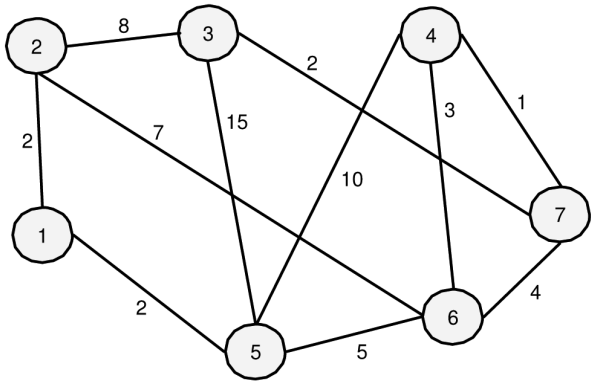
PASSO 6: Scelgo il nodo 7, lo aggiungo all'insieme S terminando il grafo



S = {3, 2, 1, 5, 6, 4, 7}

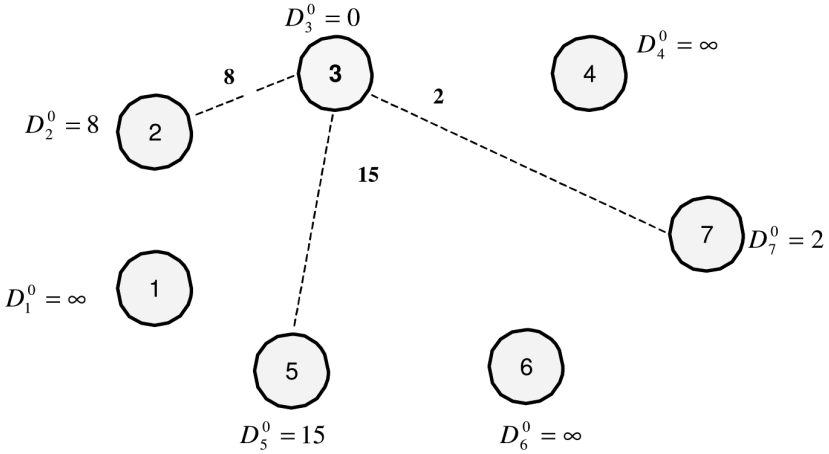
Esercizio 2 *(1° Itinere del 24/11/2003)*

Sia dato il grafo *G=(N,A)* pesato e non orientato riportato nella pagina seguente. Utilizzando l'algoritmo di Dijkstra, calcolare i percorsi minimi da qualunque nodo del grafo al nodo 3 (destinatario). Indicare con precisione ogni passo iterativo dell'algoritmo.



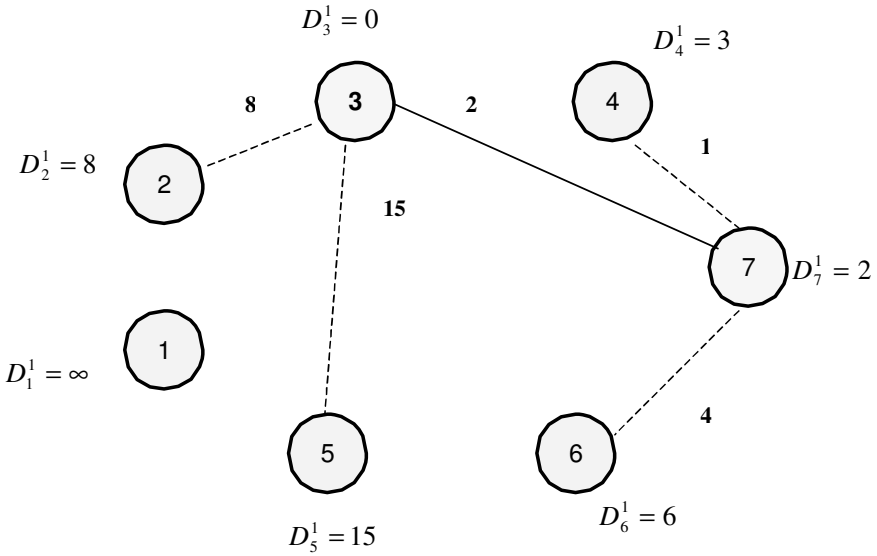
Soluzione

Passo 0: valuto i nodi 2 – 5 – 7 collegati con il nodo 3:



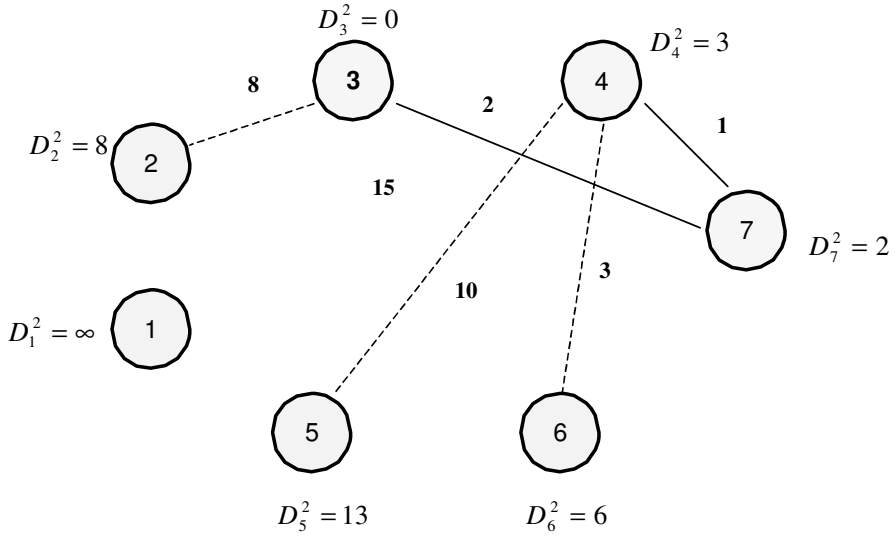
S = {3}

PASSO 1: scelgo il nodo 7, lo aggiungo all'insieme S e valuto i nodi 4 – 6 ad esso collegati



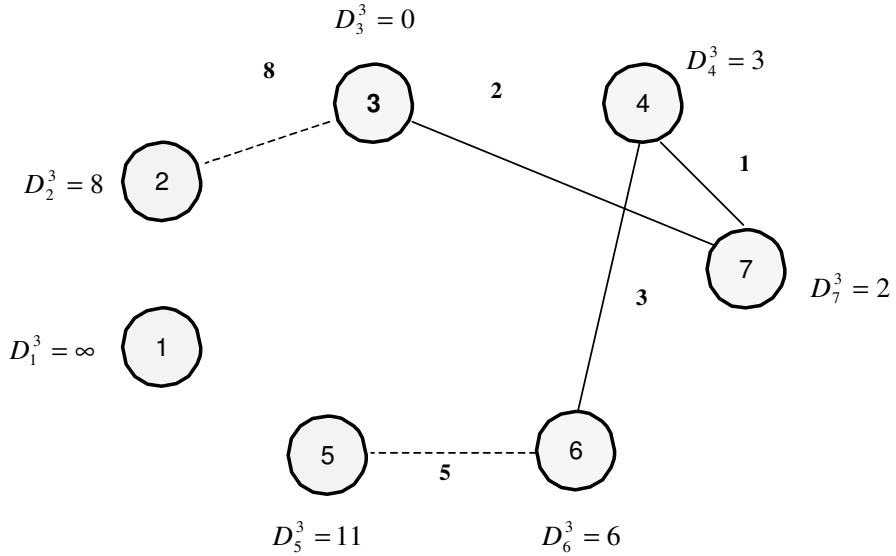
S = {3, 7}

PASSO 2: aggiungo il nodo 4 all'insieme S e valuto i nodi 6 – 5 ad esso collegati.



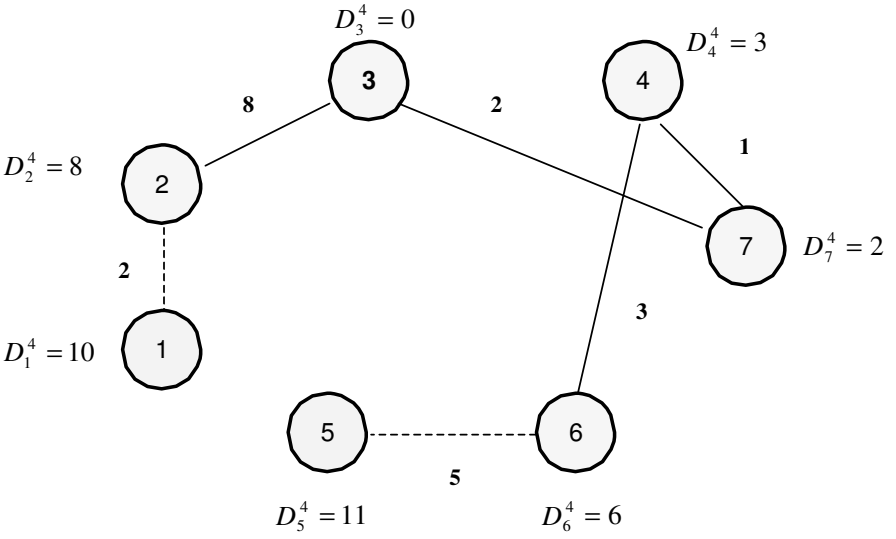
S = {3, 7, 4}

PASSO 3: aggiungo il nodo 6 all'insieme S e valuto i nodi 5 – 2 a lui collegati.



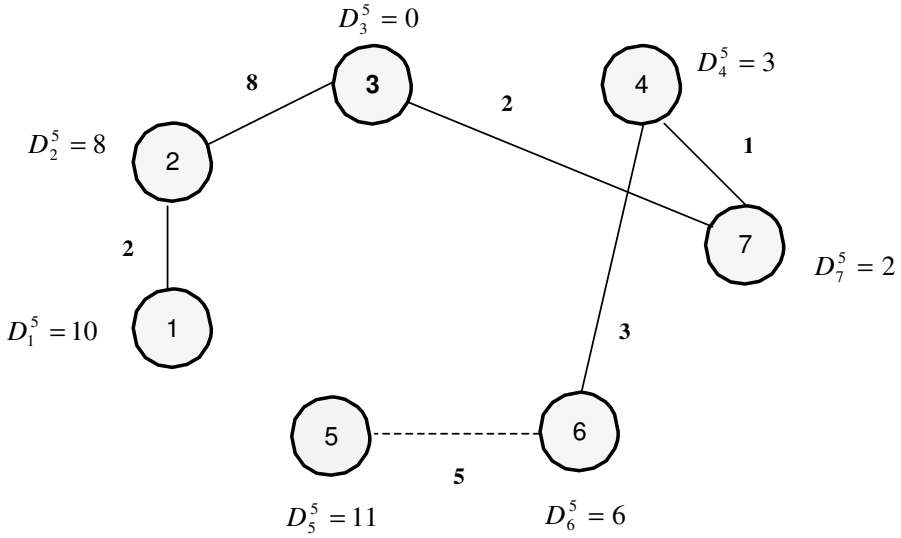
S = {3, 7, 4, 6}

PASSO 4: aggiungo il nodo 2 all'insieme S e valuto il nodo 1 a lui collegato



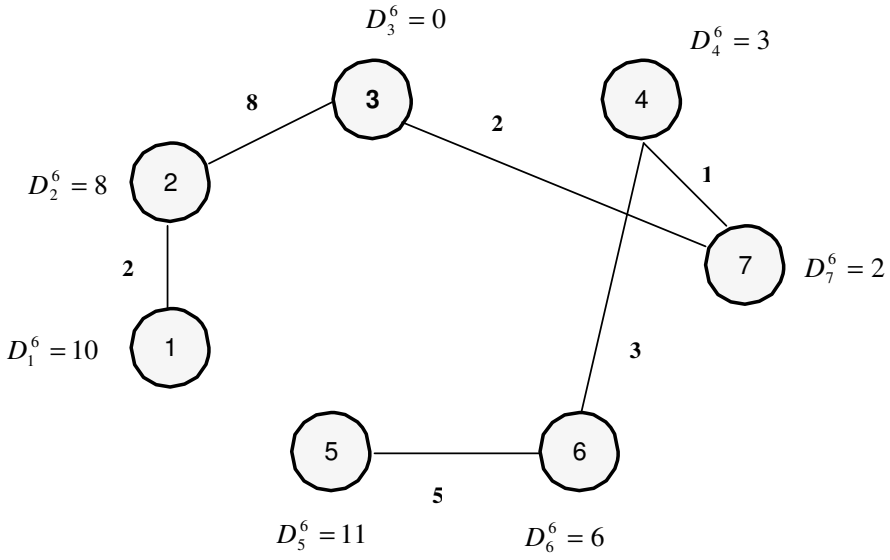
S ={3, 7, 4, 6, 2}

PASSO 5: aggiungo il nodo 2 all'insieme S e valuto il nodo 1.



S = {3, 7, 4, 6, 2, 1}

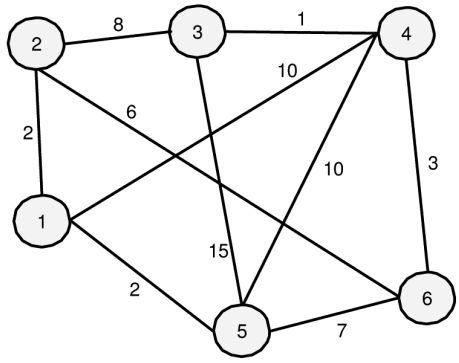
PASSO 6: aggiungo il nodo 5 all'insieme S e ho finito di collegare tutti i nodi.



S = {3, 7, 4, 6, 2, 1, 5}

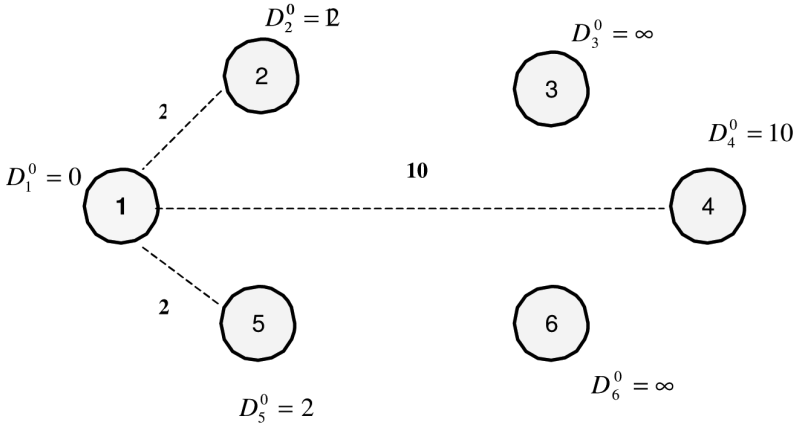
Esercizio 3 *(Appello del 10/02/2004)*

Sia dato il grafo *G=(N,A)* pesato e non orientato riportato nella pagina seguente. Utilizzando l'algoritmo di Dijkstra, calcolare i percorsi minimi da qualunque nodo del grafo al nodo 1 (destinatario). Indicare con precisione ogni passo iterativo dell'algoritmo.



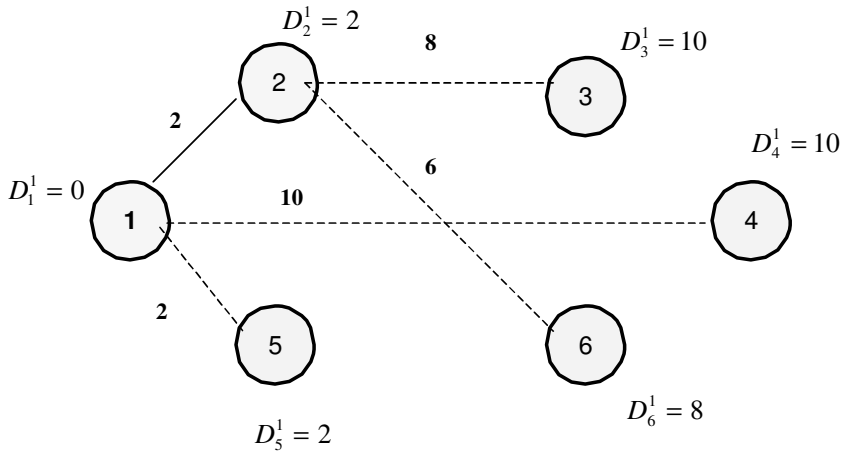
Soluzione

PASSO 0: valuto i nodi 2 – 4 -5 collegati con il nodo 1



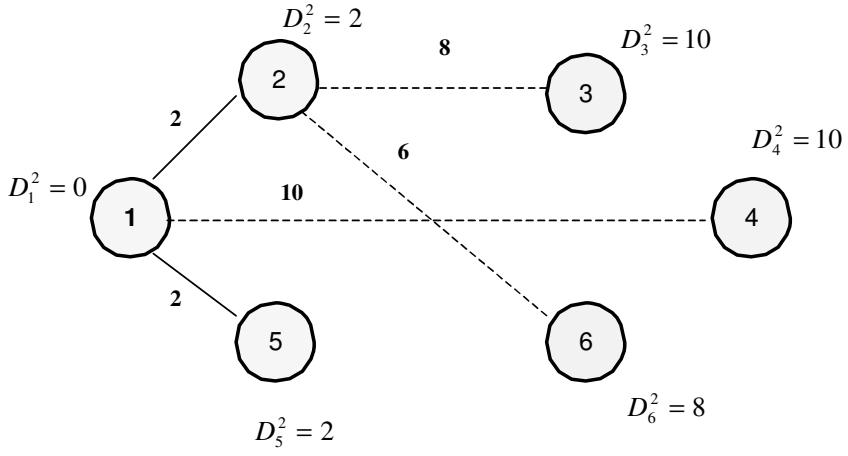
S = {1}

PASSO 1: Potrei scegliere sia il nodo 2 che il nodo 5. Scelgo il nodo 2, lo aggiungo all'insieme S e valuto i nodi 3 - 6 ad esso collegati



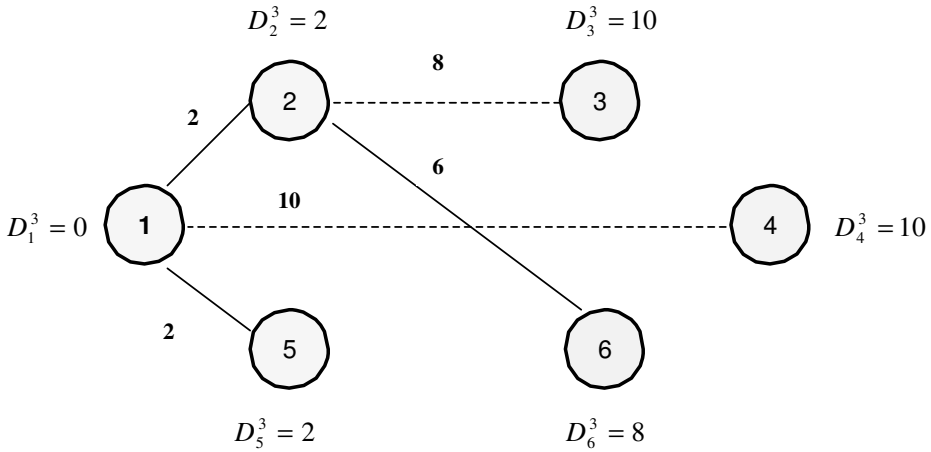
S = {1,2}

PASSO 2: Scelgo il nodo 5, lo aggiungo all'insieme S.



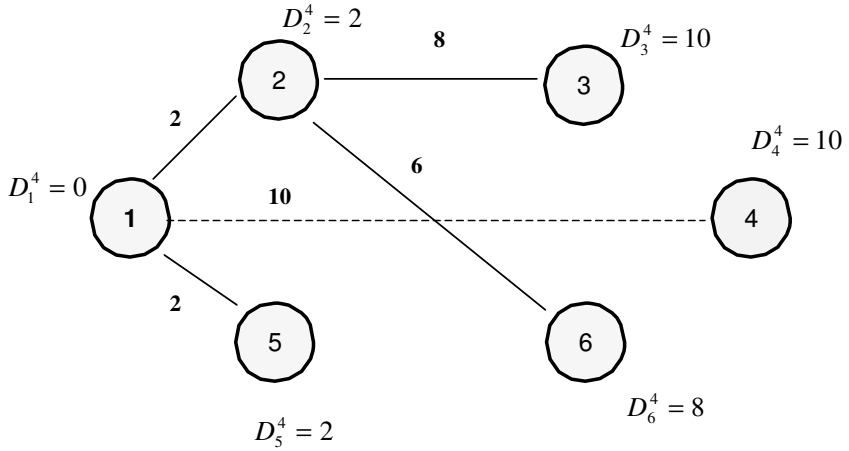
S = {1,2,5}

PASSO 3: Scelgo il nodo 6, lo aggiungo all'insieme S. Valuto quindi i nodi collegati al nodo 5.



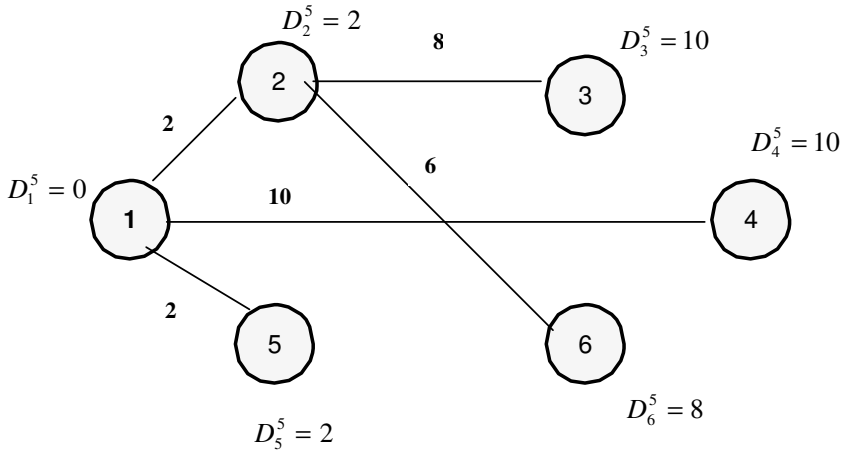
S = {1,2,5,6}

PASSO 4: Scelgo il nodo 3, lo aggiungo all'insieme S.



S = {1,2,5,6,3}

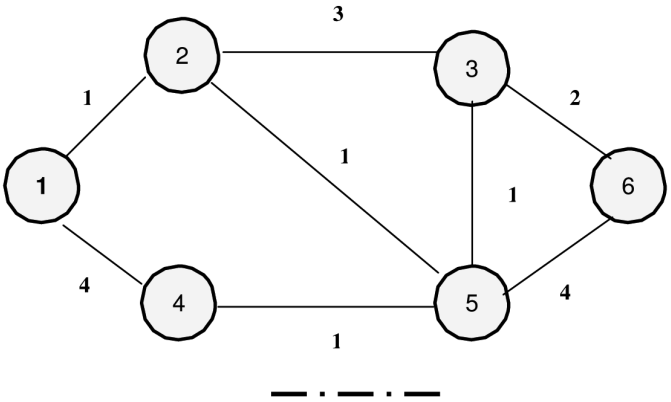
PASSO 5: Scelgo il nodo 4, lo aggiungo all'insieme S.



S = {1,2,5,6,3,4}

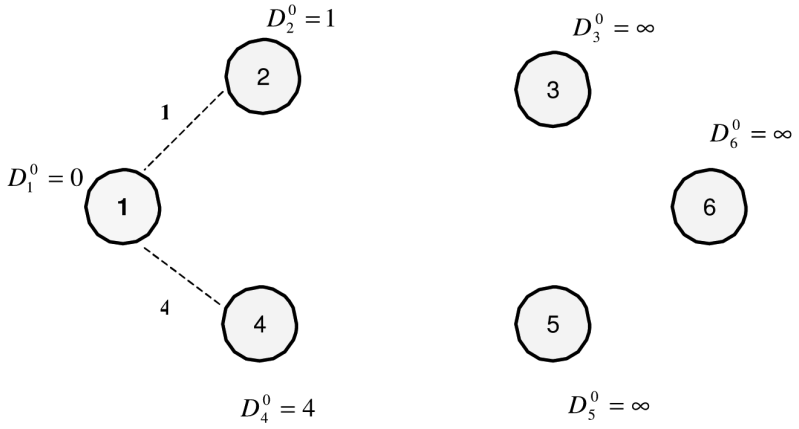
Esercizio 4 *(1° Itinere del 30/04/2002)*

Sia dato il grafo pesato e non orientato riportato in figura. Utilizzando l'algoritmo di Dijkstra, calcolare i percorsi minimi da qualunque nodo del grafo al nodo 1 (destinatario).



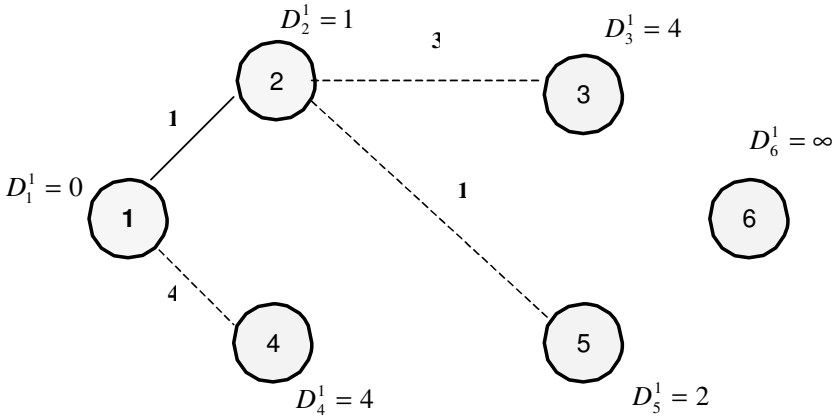
Soluzione

PASSO 0: valuto i nodi 2 – 4 collegati con il nodo 1



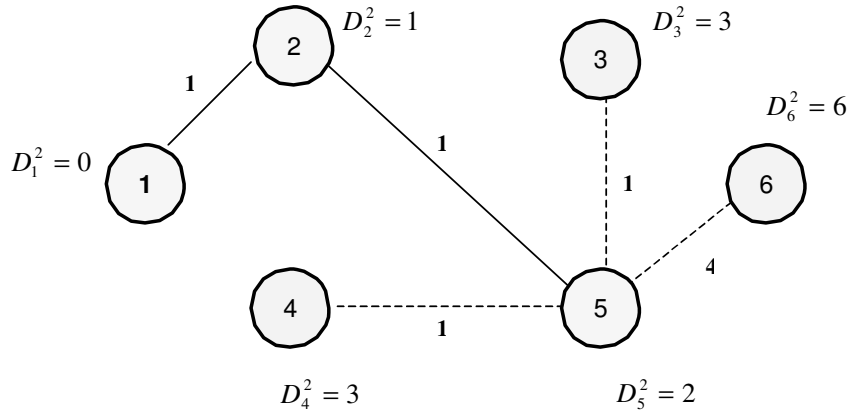
S={1}

PASSO 1: Scelgo il nodo 2, lo aggiungo all'insieme S e valuto i nodi 3 - 5 ad esso collegati e il nodo 4



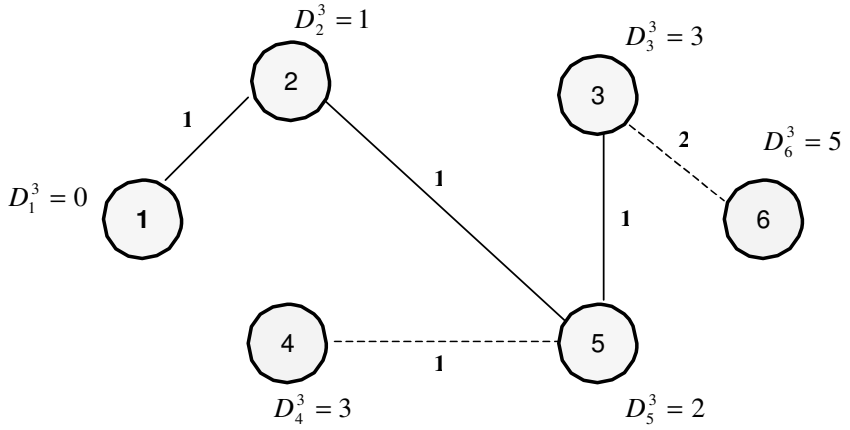
S={1,2}

PASSO 2: Scelgo il nodo 5, lo aggiungo all'insieme S e valuto i nodi 3 - 4 ad esso collegati



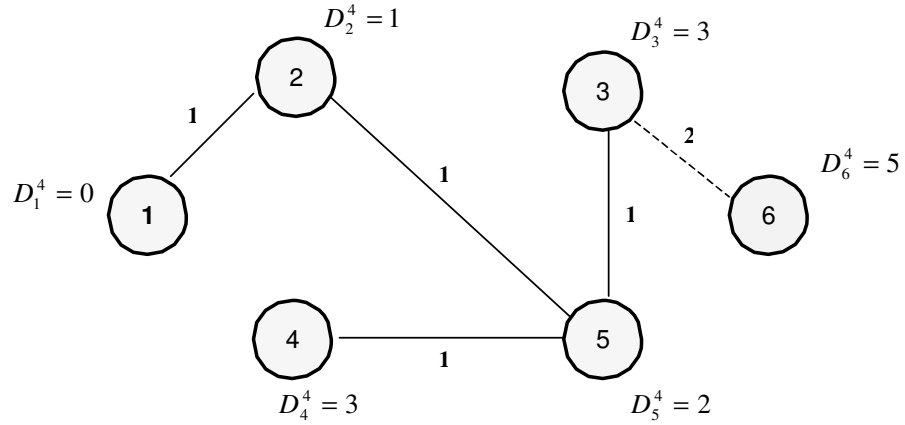
S={1,2,5}

PASSO 3: Scelgo il nodo 3, lo aggiungo all'insieme S e valuto il nodo 4 e il nodo 6 ad esso collegato



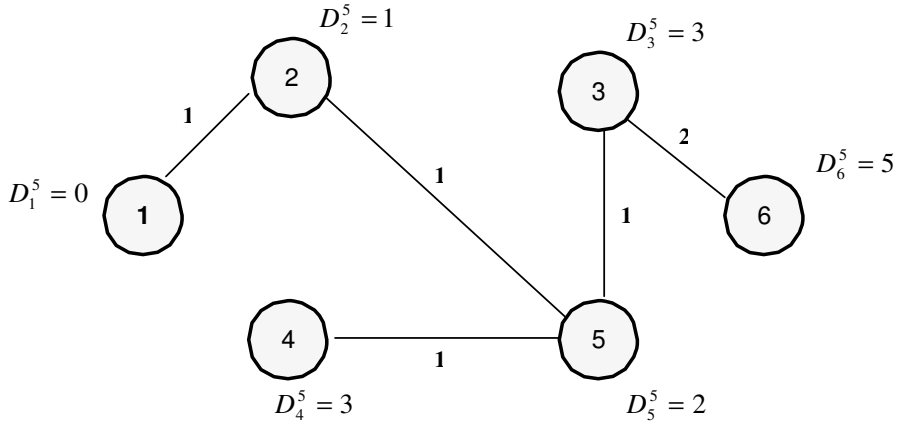
S={1,2,3,5}

PASSO 4: Scelgo il nodo 4, lo aggiungo all'insieme S e valuto il nodo 6



S={1,2,3,4,5}

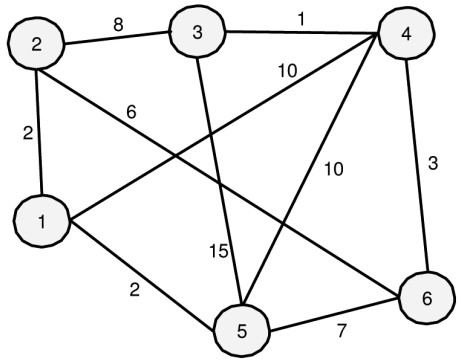
PASSO 5: Scelgo il nodo 6 e lo aggiungo all'insieme S; tutti i nodi sono stati permanentemente etichettati: l'algoritmo è terminato.



S={1,2,3,4,5,6}

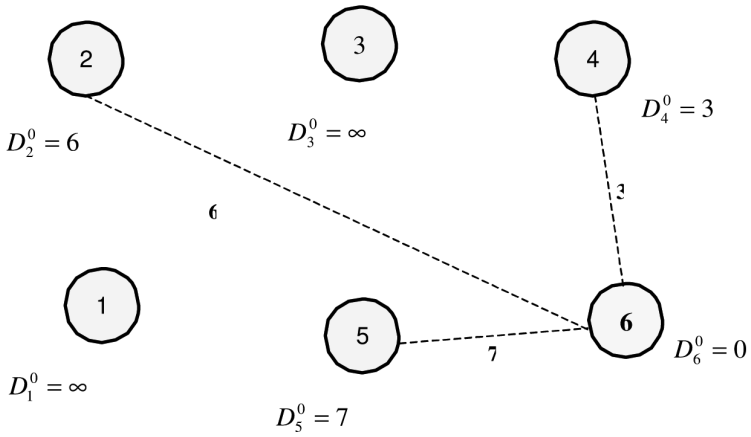
Esercizio 5 *(Appello del 25/02/2004)*

Sia dato il grafo *G=(N,A)* pesato e non orientato riportato nella pagina seguente. Utilizzando l'algoritmo di Dijkstra, calcolare i percorsi minimi da qualunque nodo del grafo al nodo 6 (destinatario). Indicare con precisione ogni passo iterativo dell'algoritmo.



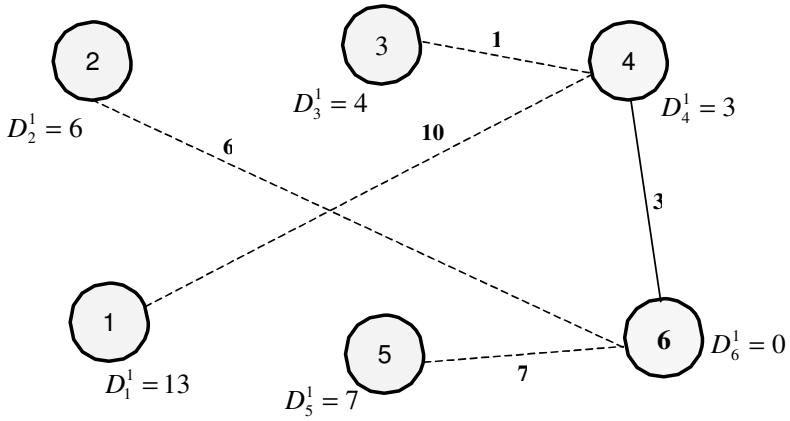
Soluzione

Passo 0: valuto i nodi 2 – 4 – 5 collegati al nodo 6



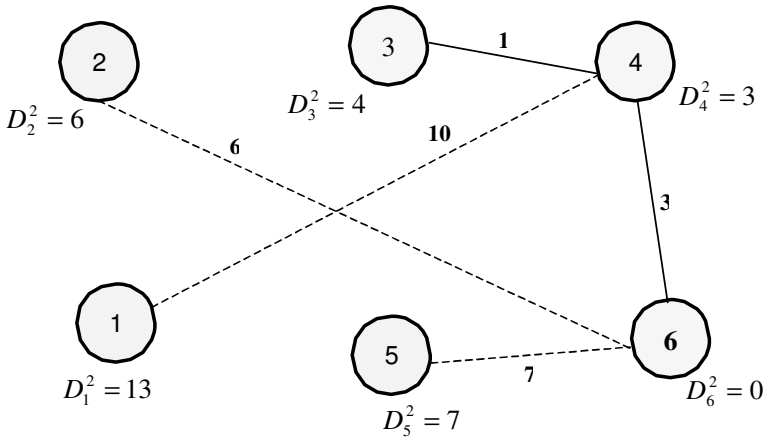
S={6}

Passo 1: scelgo il nodo 4 poiché il più vicino, lo aggiungo all'insieme S e valuto i nodi 1 – 3 ad esso collegati



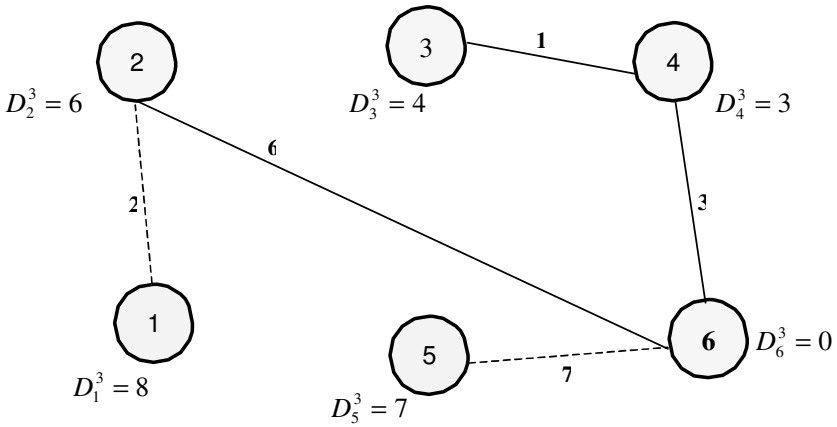
S={6, 4}

Passo 2: scelgo il nodo 3, lo aggiungo all'insieme S e valuto i nodi ad esso collegati: tutti i nodi hanno minor distanza da 6 se non si passa attraverso il nodo 3



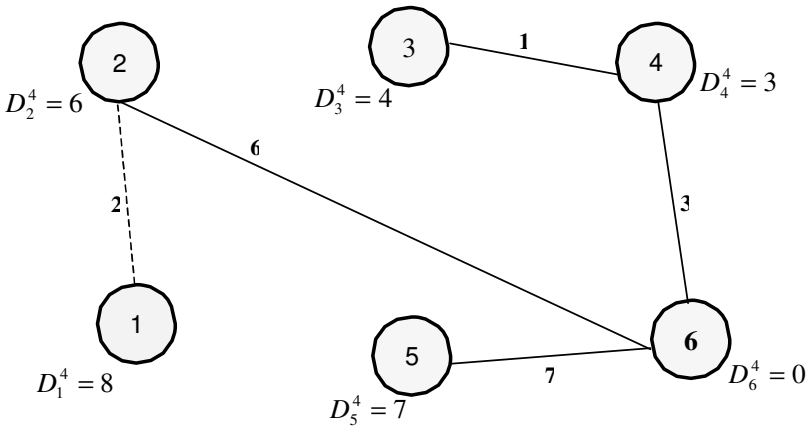
S={6, 4, 3}

Passo 3: scelgo il nodo 6 perché il più vicino al nodo di destinazione, lo aggiungo all'insieme S e valuto il nodo 1 ad esso collegato



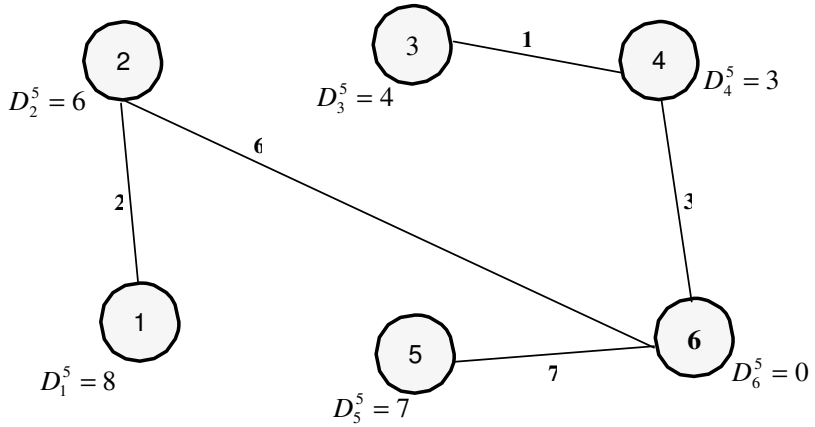
S={6, 4, 3, 2}

Passo 4: collego il nodo 5, lo aggiungo all'insieme S



S={6, 4, 3, 2, 5}

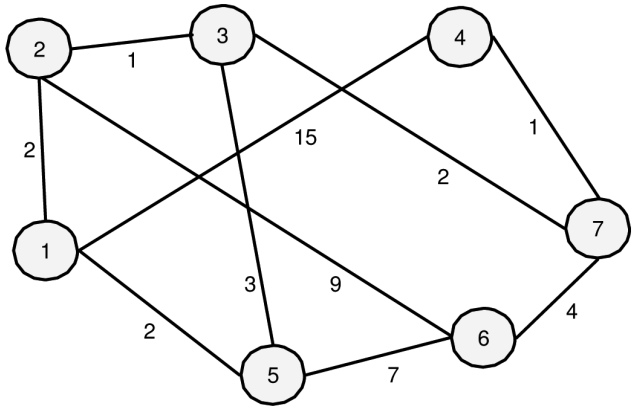
Passo 5 (passo finale): collego il nodo 1 e lo aggiungo all'insieme S



S={6, 4, 3, 2, 5, 1}

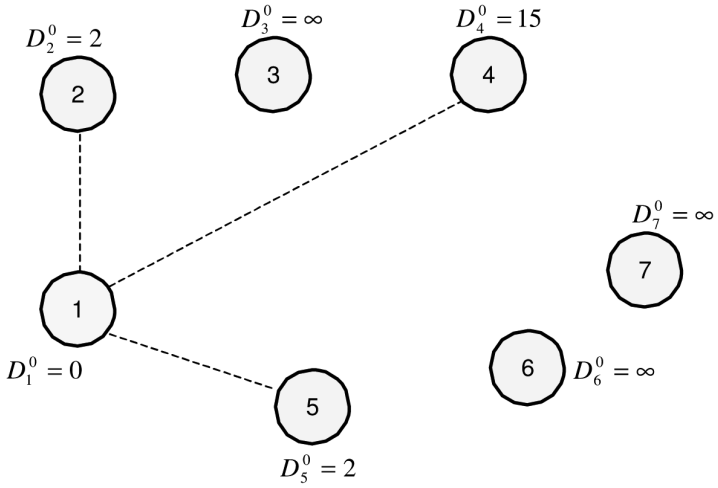
Esercizio 6 *(Appello del 25/02/2003)*

Sia dato il grafo *G=(N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Dijkstra, calcolare il percorso a costo minimo da ogni nodo di *N* al nodo 1 (destinatario). Indicare con rigore i vari passi dell'algoritmo, utilizzando, se possibile, le notazioni usate a lezione.



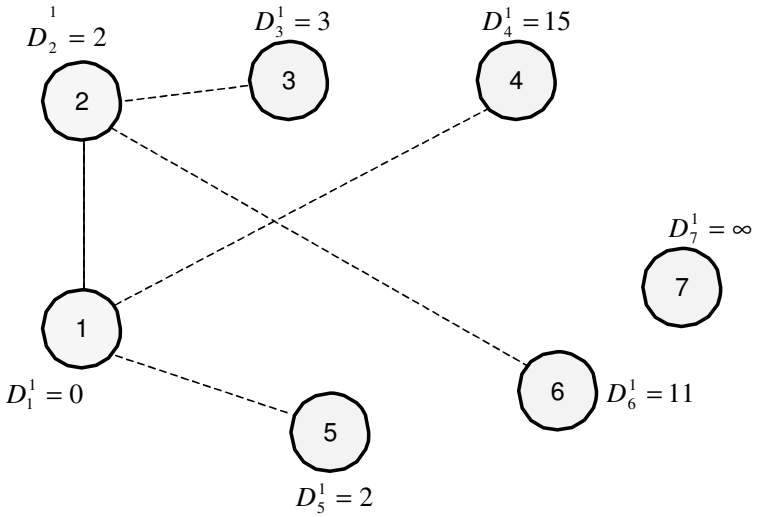
Soluzione

PASSO 0: valuto i nodi 2, 5 e 4 collegati al nodo 1, inserisco come primo elemento nell'insieme S il nodo destinatario che ha distanza uguale a 0.



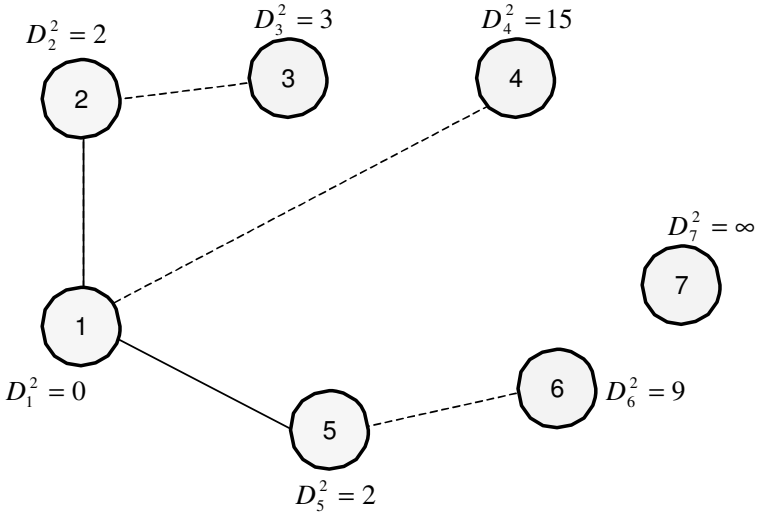
S = {1}

PASSO 1: scelgo il nodo 2, lo aggiungo all'insieme S e valuto i nodi 3 e 6 ad esso collegati



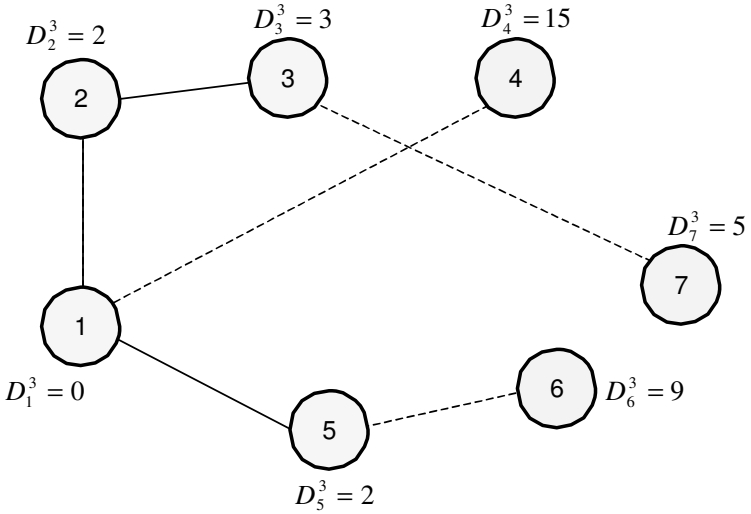
S = {1,2}

PASSO 2: scelgo il nodo 5, lo aggiungo all'insieme S e valuto il nodo 6 ad esso collegato, non considero il collegamento 5-3 perché risulterebbe più costoso di $D_3^1 = 3$



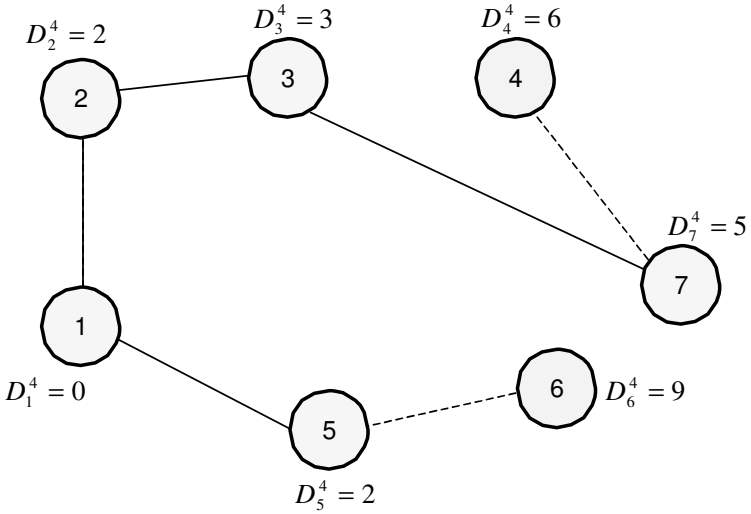
S = {1,2,5}

PASSO 3: scelgo il nodo 3, lo aggiungo all'insieme S e valuto il nodo 7 ad esso collegato



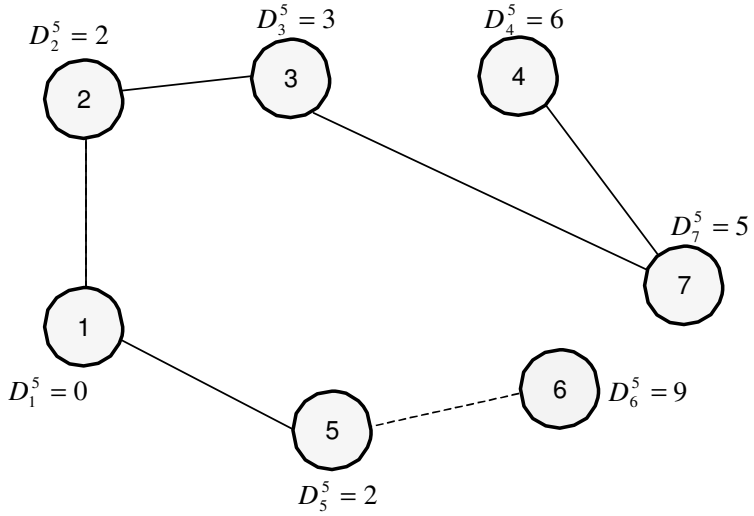
S = {1,2,5,3}

PASSO 4: scelgo il nodo 7, lo aggiungo all'insieme S e valuto il nodo 4 ad esso collegato, non considero il collegamento 7-6 perché risulterebbe ugualmente costoso a $D_6^3 = 9$



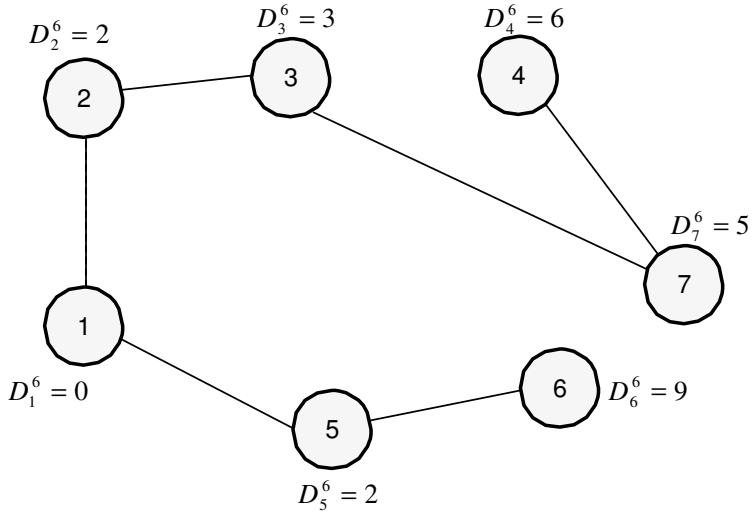
S = {1,2,5,3,7}

PASSO 5: scelgo il nodo 4, lo aggiungo all'insieme S, non vi sono nodi ad esso collegati da considerare.



S = {1,2,5,3,7,4}

PASSO 5: scelgo il nodo 6, lo aggiungo all'insieme S, non vi sono altri nodi da considerare, abbiamo ottenuto il percorso a costo minimo.

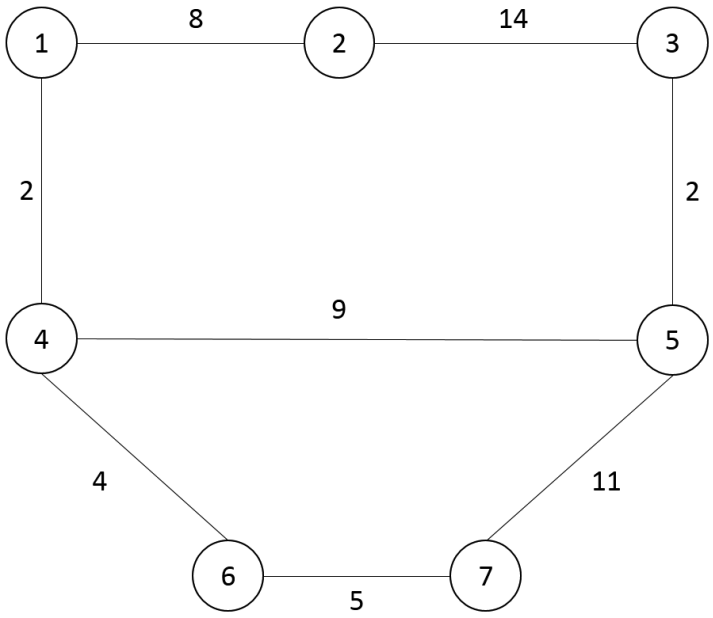


S = {1,2,5,3,7,4,6}

CAPITOLO 2: INSTRADAMENTO CON ALGORITMO DI BELLMAN-FORD

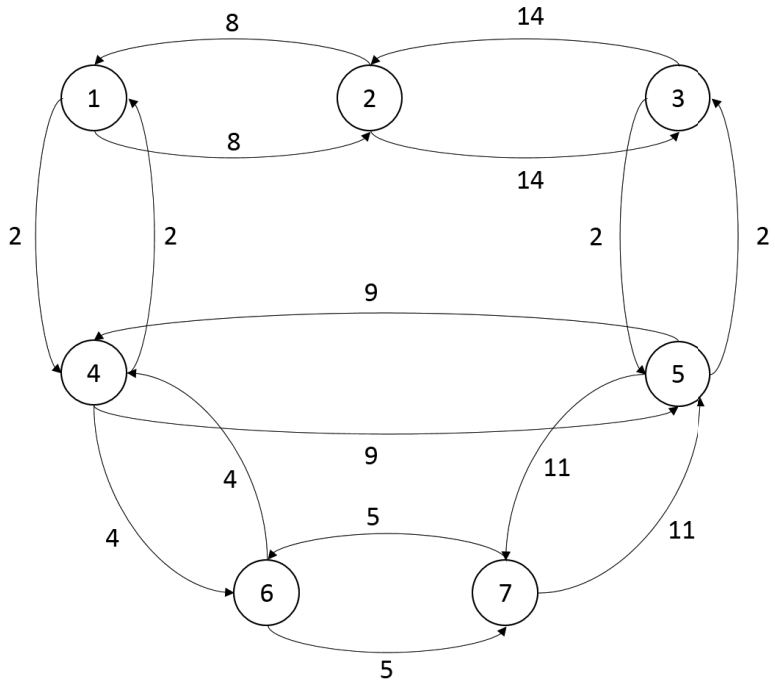
Esercizio 1

Sia dato il grafo *G= (N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Bellman-Ford, calcolare i percorsi a costo minimo dal nodo 3 ad ogni nodo di *N*. Indicare con rigore i vari passi dell'algoritmo.



Soluzione

L'algoritmo di Bellman-Ford è definito per grafi orientati. Per questo motivo, costruiamo il grafo orientato associato a quello assegnato.



Passo 0

Inizializziamo l'insieme L degli archi dell'albero dei cammini minimi dal nodo 3 agli altri nodi di N, il costo associato ai nodi, D_i, e gli archi utilizzati per costruire i cammini minimi associati ai vari nodi h_i.

$$L = \emptyset, D_3 = 0, D_i = \infty \quad \forall\, i \in N - \{3\}, h_i = null \quad \forall\, i \in N$$

Elenchiamo gli archi per facilitare l'aggiornamento dei pesi (l'ordine è scelto a piacere):

a_{32}
a_{35}
a_{21}
a_{14}
a_{45}
a_{46}
a_{67}
a_{57}
a_{23}
a_{53}
a_{12}
a_{41}
a_{54}
a_{64}
a_{76}
a_{75}

Passiamo ad aggiornare i pesi dei nodi, D_i, per ogni arco, per un numero di volte massimo pari a $|N|$-1=6 volte.

Passo 1 (aggiornamento #1 di 6)

$a_{32}, D_2 = \min[D_2, D_3 + d_{32}] = \min[\infty, 0 + 14] = 14, h_2 = a_{32}$
$a_{35}, D_5 = 2, h_5 = a_{35}$
$a_{21}, D_1 = 22, h_1 = a_{21}$
$a_{14}, D_4 = 24, h_4 = a_{14}$
$a_{45}, D_5 = 2, \text{non aggiornato}$
$a_{46}, D_6 = 28, h_6 = a_{46}$
$a_{67}, D_7 = 33, h_7 = a_{67}$
$a_{57}, D_7 = 13, h_7 = a_{57}$
$a_{23}, D_3 = 0, \text{non aggiornato}$
$a_{53}, D_3 = 0, \text{non aggiornato}$
$a_{12}, D_2 = 14, \text{non aggiornato}$
$a_{41}, D_1 = 22, \text{non aggiornato}$
$a_{54}, D_4 = 11, h_4 = a_{54}$
$a_{64}, D_4 = 11, \text{non aggiornato}$
$a_{76}, D_6 = 18, h_6 = a_{76}$
$a_{75}, D_5 = 2, \text{non aggiornato}$

Rappresentiamo le informazioni ottenute alla fine di questo passo iterativo nella seguente tabella:

	1	**2**	**3**	**4**	**5**	**6**	**7**
D	22	14	0	11	2	18	13
h	a_{21}	a_{32}	/	a_{54}	a_{35}	a_{76}	a_{57}

Passo 2 (aggiornamento #2 di 6)

$a_{32}, D_2 = 14, \text{non aggiornato}$
$a_{35}, D_5 = 2, \text{non aggiornato}$
$a_{21}, D_1 = 22, \text{non aggiornato}$
$a_{14}, D_4 = 11, \text{non aggiornato}$
$a_{45}, D_5 = 2, \text{non aggiornato}$
$a_{46}, D_6 = 15, h_6 = a_{46}$
$a_{67}, D_7 = 13, \text{non aggiornato}$
$a_{57}, D_7 = 13, \text{non aggiornato}$
$a_{23}, D_3 = 0, \text{non aggiornato}$
$a_{53}, D_3 = 0, \text{non aggiornato}$
$a_{12}, D_2 = 14, \text{non aggiornato}$
$a_{41}, D_1 = 13, h_1 = a_{41}$
$a_{54}, D_4 = 11, \text{non aggiornato}$
$a_{64}, D_4 = 11, \text{non aggiornato}$
$a_{76}, D_6 = 15, \text{non aggiornato}$
$a_{75}, D_5 = 2, \text{non aggiornato}$

Rappresentiamo le informazioni ottenute alla fine di questo passo iterativo nella seguente tabella:

	1	**2**	**3**	**4**	**5**	**6**	**7**
D	13	14	0	11	2	15	13
h	a_{41}	a_{32}	/	a_{54}	a_{35}	a_{46}	a_{57}

Passo 3 (aggiornamento #3 di 6)

$a_{32}, D_2 = 14, \text{non aggiornato}$
$a_{35}, D_5 = 2, \text{non aggiornato}$
$a_{21}, D_1 = 22, \text{non aggiornato}$
$a_{14}, D_4 = 11, \text{non aggiornato}$
$a_{45}, D_5 = 2, \text{non aggiornato}$
$a_{46}, D_6 = 15, \text{non aggiornato}$
$a_{67}, D_7 = 13, \text{non aggiornato}$
$a_{57}, D_7 = 13, \text{non aggiornato}$
$a_{23}, D_3 = 0, \text{non aggiornato}$
$a_{53}, D_3 = 0, \text{non aggiornato}$
$a_{12}, D_2 = 14, \text{non aggiornato}$
$a_{41}, D_1 = 13, , \text{non aggiornato}$
$a_{54}, D_4 = 11, \text{non aggiornato}$
$a_{64}, D_4 = 11, \text{non aggiornato}$

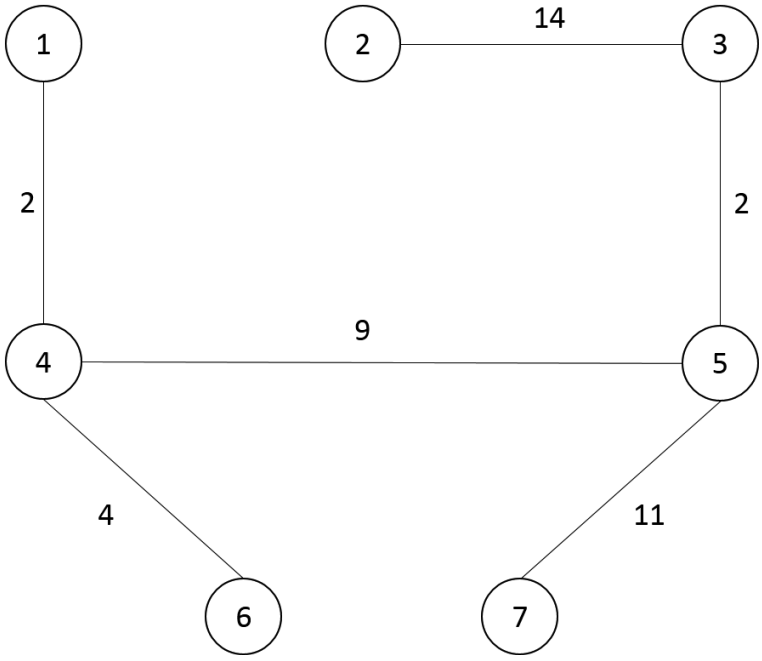
$a_{76}, D_6 = 15, \text{non aggiornato}$
$a_{75}, D_5 = 2, \text{non aggiornato}$

Non c'è stato nessun aggiornamento dei pesi degli archi, di conseguenza l'algoritmo di Bellman-Ford si ferma. Non è necessario riscrivere la tabella visto che coincide con quella del passo 2.

Partendo dall'ultima tabella (quella del passo 2), definiamo l'albero dei cammini minimi dal nodo 3 agli altri nodi del grafo. L'insieme degli archi dell'albero, *L*, è dato dall'insieme di archi specificato nella tabella.

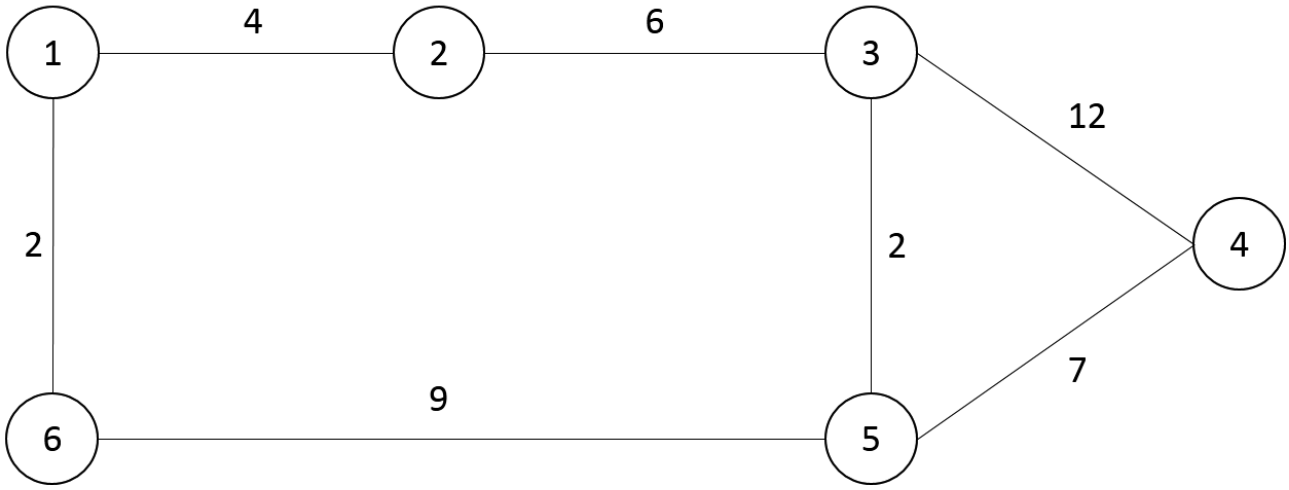
$L = \{a_{41}, a_{32}, a_{54}, a_{35}\ , a_{46}, a_{57}\ \}$

L'albero è dunque il seguente *T=(N, L)*, rappresentato anche in figura (gli archi sono rappresentati come non orientati visto che nel grafo reale di partenza non avevano orientamento; l'orientamento è stato introdotto solo per applicare Bellman-Ford).



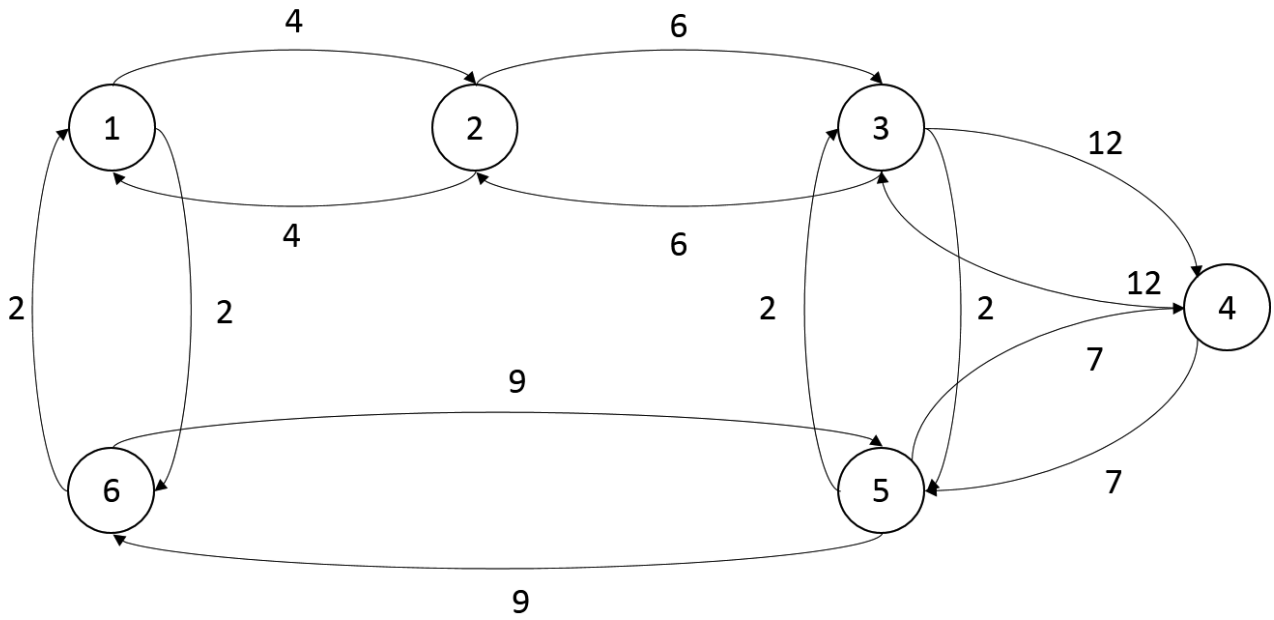
Esercizio 2

Sia dato il grafo *G= (N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Bellman-Ford, calcolare i percorsi a costo minimo dai nodi 3 e 6 verso il nodo 1. Indicare con rigore i vari passi dell'algoritmo.



Soluzione

L'algoritmo di Bellman-Ford è definito per grafi orientati. Per questo motivo, costruiamo il grafo orientato associato a quello assegnato.



Nel nostro caso si richiede di trovare i percorsi minimi dai nodi 3 e 6 verso il nodo 1. Dato che il grafo iniziale è non orientato, l'albero dei percorsi a costo minimo dai nodi 3 e 6 verso il nodo 1 coincide con l'albero dei percorsi a costo minimo dal nodo 1 verso i nodi 3 e 6. Per tale motivo, applichiamo Bellman-Ford come se dovessimo trovare i percorsi minimi aventi come origine il nodo 1. Inoltre, anche se in realtà dobbiamo trovare

solo i percorsi minimi da 1 verso 3 e 6, applichiamo Bellman-Ford come se dovessimo trovare i percorsi minimi verso tutti i nodi *N*.

Passo 0

Inizializzo l'insieme *L* degli archi dell'albero dei cammini minimi dal nodo 1 verso i nodi 3 e 6, il costo associato ai nodi, D_i, e gli archi utilizzati per costruire i cammini minimi associati ai vari nodi h_i.

$$L = \emptyset, D_1 = 0, D_i = \infty \quad \forall\, i \in N - \{1\}, h_i = null \quad \forall\, i \in N$$

Elenchiamo gli archi per velocizzare l'aggiornamento dei pesi (l'ordine è definito a piacere):

a_{12}
a_{16}
a_{23}
a_{65}
a_{35}
a_{34}
a_{45}
a_{21}
a_{61}
a_{32}
a_{56}
a_{53}
a_{43}
a_{54}

Passiamo ad aggiornare i pesi dei nodi, D_i, per ogni arco, per un numero di volte massimo pari a |*N*|-1=5.

Passo 1 (aggiornamento #1 di 5)

$a_{12}, D_2 = \min[D_2, D_1 + d_{12}] = \min[\infty, 0 + 4] = 4, h_2 = a_{12}$
$a_{16}, D_6 = 2, h_6 = a_{16}$
$a_{23}, D_3 = 10, h_3 = a_{23}$
$a_{65}, D_5 = 11, h_5 = a_{65}$
$a_{35}, D_5 = 11, \text{non aggiornato}$
$a_{34}, D_4 = 22, h_4 = a_{34}$
$a_{45}, D_5 = 11, \text{non aggiornato}$
$a_{21}, D_1 = 0, \text{non aggiornato}$
$a_{61}, D_1 = 0, \text{non aggiornato}$
$a_{32}, D_2 = 4, \text{non aggiornato}$
$a_{56}, D_6 = 2, \text{non aggiornato}$
$a_{53}, D_3 = 10, \text{non aggiornato}$
$a_{43}, D_3 = 10, \text{non aggiornato}$
$a_{54}, D_4 = 18, h_4 = a_{54}$

Rappresentiamo le informazioni ottenute alla fine di questo passo iterativo nella seguente tabella:

	1	**2**	**3**	**4**	**5**	**6**
D	0	4	10	18	11	2
h	/	a_{12}	a_{23}	a_{54}	a_{65}	a_{16}

Passo 2 (aggiornamento #2 di 5)

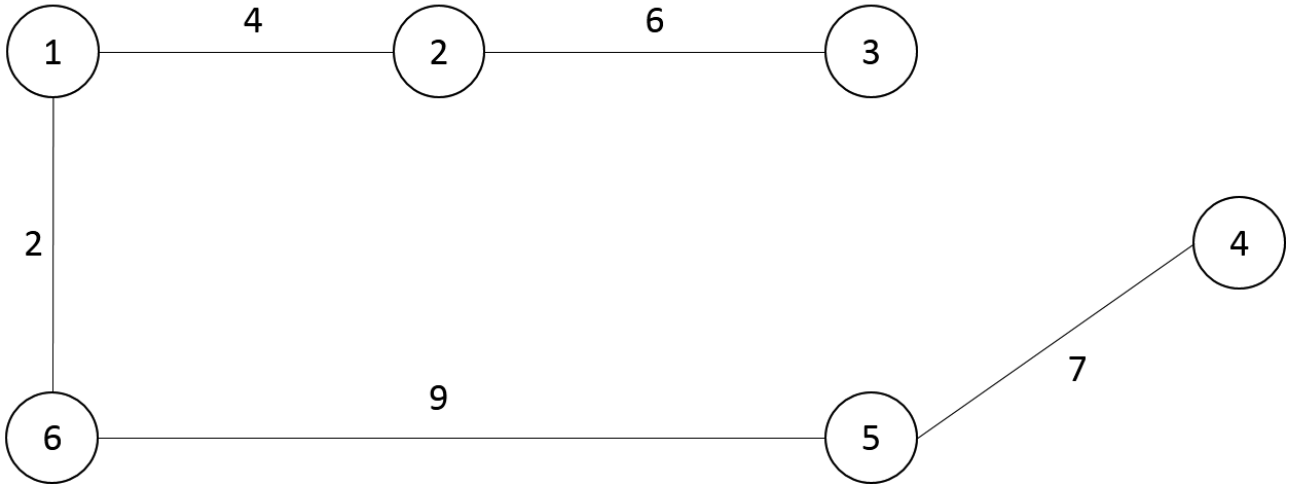
$a_{12}, D_2 = 4, \text{non aggiornato}$
$a_{16}, D_6 = 2, \text{non aggiornato}$
$a_{23}, D_3 = 10, \text{non aggiornato}$
$a_{65}, D_5 = 11, \text{non aggiornato}$
$a_{35}, D_5 = 11, \text{non aggiornato}$
$a_{34}, D_4 = 18, \text{non aggiornato}$
$a_{45}, D_5 = 11, \text{non aggiornato}$
$a_{21}, D_1 = 0, \text{non aggiornato}$
$a_{61}, D_1 = 0, \text{non aggiornato}$
$a_{32}, D_2 = 4, \text{non aggiornato}$
$a_{56}, D_6 = 2, \text{non aggiornato}$
$a_{53}, D_3 = 10, \text{non aggiornato}$
$a_{43}, D_3 = 10, \text{non aggiornato}$
$a_{54}, D_4 = 18, \text{non aggiornato}$

Non c'è stato nessun aggiornamento dei pesi degli archi, di conseguenza l'algoritmo di Bellman-Ford si ferma. Non è necessario riscrivere la tabella visto che coincide con quella del passo 1.

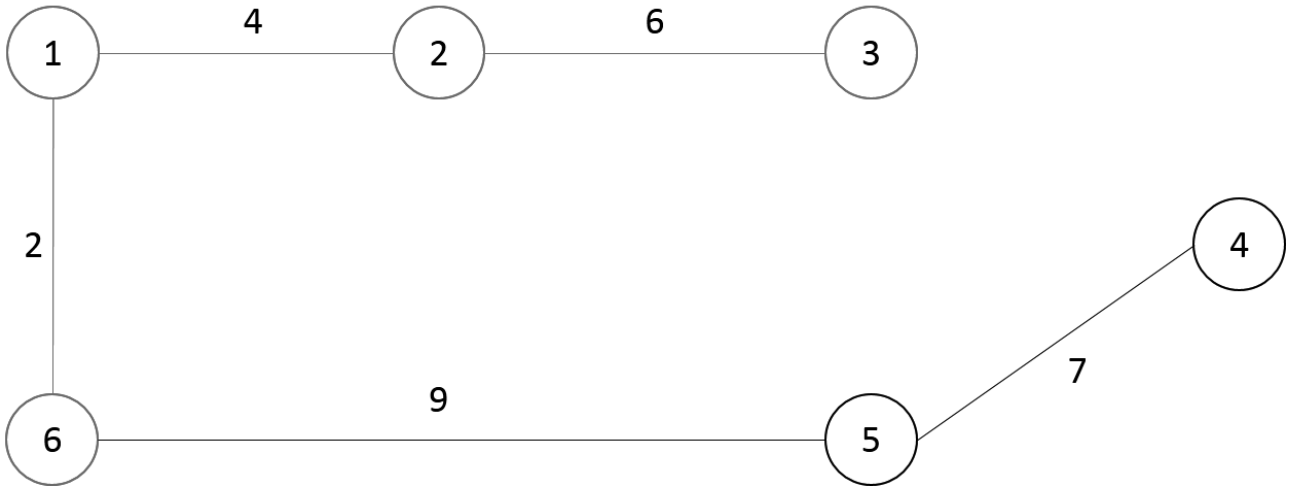
Partendo dall'ultima tabella (quella del passo 1), definiamo l'albero dei cammini minimi ottenuto. L'insieme degli archi dell'albero, *L*, è dato dall'insieme di archi specificato nella tabella.

$L = \{a_{12}, a_{23}, a_{54}, a_{65}, a_{16}\}$

L'albero è dunque il seguente *T=(N, L)*, rappresentato anche in figura (gli archi sono rappresentati come non orientati visto che nel grafo reale di partenza non avevano orientamento; l'orientamento è stato introdotto solo per applicare Bellman-Ford).



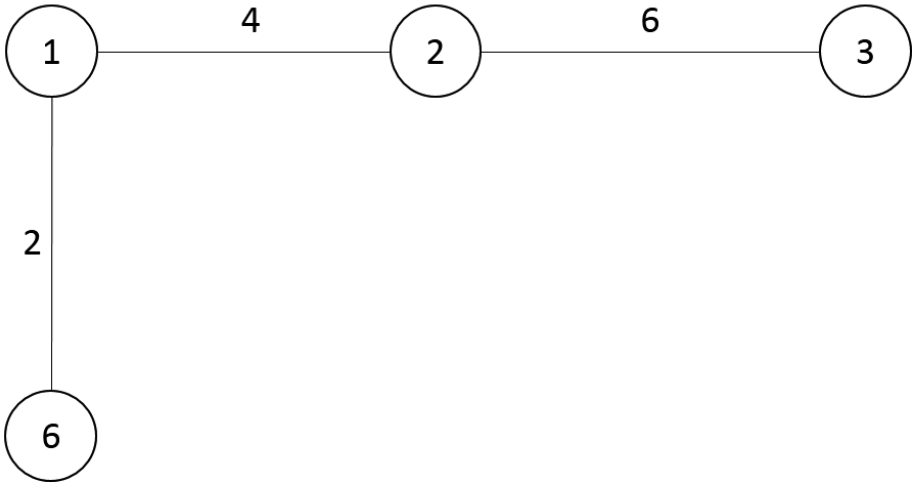
Tale albero rappresenta l'albero dei cammini minimi da 1 verso tutti gli altri nodi. A noi, però, servono solo i cammini minimi verso i nodi 3 e 6. A tal fine, partendo dall'albero trovato, evidenziamo gli archi che ci servono per raggiungere dal nodo 1 i nodi 3 e 6.



Come si vede, i nodi 5 e 4 non sono richiesti per raggiungere 3 e 6, così come gli archi a_{65}e a_{54}. Per tale motivo, l'albero dei cammini minimi da 1 ai nodi 3 e 6 è il grafo T'= (N', L'), rappresentato in figura, dove:

$N' = \{1,2,3,6\}$

$L' = \{a_{12}, a_{23}, a_{16}\}$

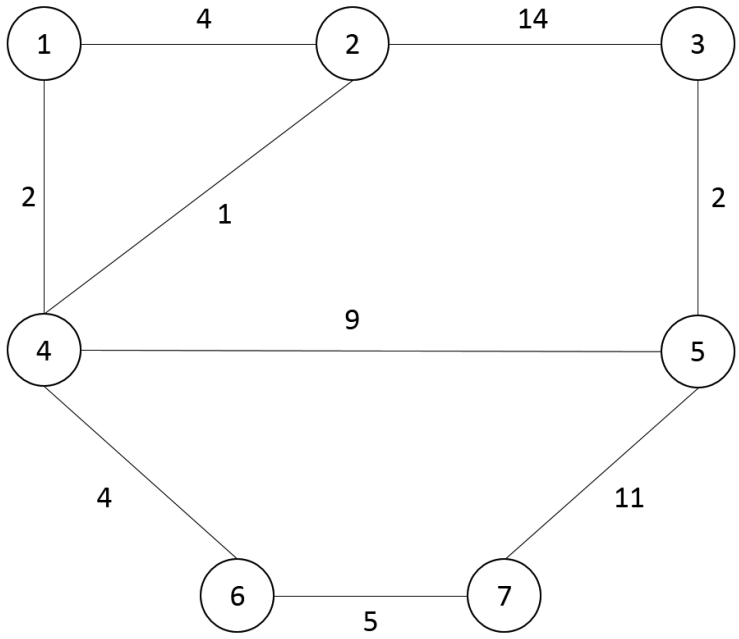


Dato che il grafo è non orientato, tale albero dei cammini mini dal nodo 1 verso i nodi 3 e 6, coincide con il grafo dei cammini minimi dai nodi 3 e 6 verso il nodo 1.

CAPITOLO 3:
INSTRADAMENTO CON ALGORITMO DI KRUSKAL

Esercizio 1

Sia dato il grafo *G= (N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Kruskal, calcolare l'albero a costo minimo che connette i nodi *N*. Indicare con rigore i vari passi dell'algoritmo e il costo della soluzione trovata.



Soluzione

Passo 0

Inizializziamo l'insieme *L* degli archi dell'albero di costo minimo.

$L = \emptyset$

Inoltre, ordiniamo gli archi del grafo in ordine crescente di costo.

$a_{24} \rightarrow 1$
$a_{14} \rightarrow 2$
$a_{35} \rightarrow 2$
$a_{12} \rightarrow 4$
$a_{46} \rightarrow 4$
$a_{67} \rightarrow 5$
$a_{45} \rightarrow 9$
$a_{57} \rightarrow 11$
$a_{23} \rightarrow 14$

Iniziamo a creare l'albero, aggiungendo gli opportuni archi sulla base della lista appena definita e fermando l'algoritmo quando |L|=|N|-1, ovvero |L|=6.

Passo 1

Selezioniamo l'arco a_{24}. Il grafo $T = (N, L \cup \{a_{24}\})$ non ha cicli, allora aggiungiamo l'arco a_{24} a L: $L = \{a_{24}\}$.

$|L|$=1 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 2

Selezioniamo l'arco a_{14}. Il grafo $T = (N, L \cup \{a_{14}\})$ non ha cicli, allora aggiungiamo l'arco a_{14} a L: $L = \{a_{24}, a_{14}\}$.

$|L|$=2 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 3

Selezioniamo l'arco a_{35}. Il grafo $T = (N, L \cup \{a_{35}\})$ non ha cicli, allora aggiungiamo l'arco a_{35} a L: $L = \{a_{24}, a_{14}, a_{35}\}$.

$|L|$=3 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 4

Selezioniamo l'arco a_{12}. Il grafo $T = (N, L \cup \{a_{12}\})$ ha cicli (ad esempio, il cammino 1-2-4-1 è un ciclo), allora non aggiungiamo l'arco a_{12} a L.

$|L|$=3 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 5

Selezioniamo l'arco a_{46}. Il grafo $T = (N, L \cup \{a_{46}\})$ non ha cicli, allora aggiungiamo l'arco a_{46} a L: $L = \{a_{24}, a_{14}, a_{35}, a_{46}\}$.

$|L|$=4 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 6

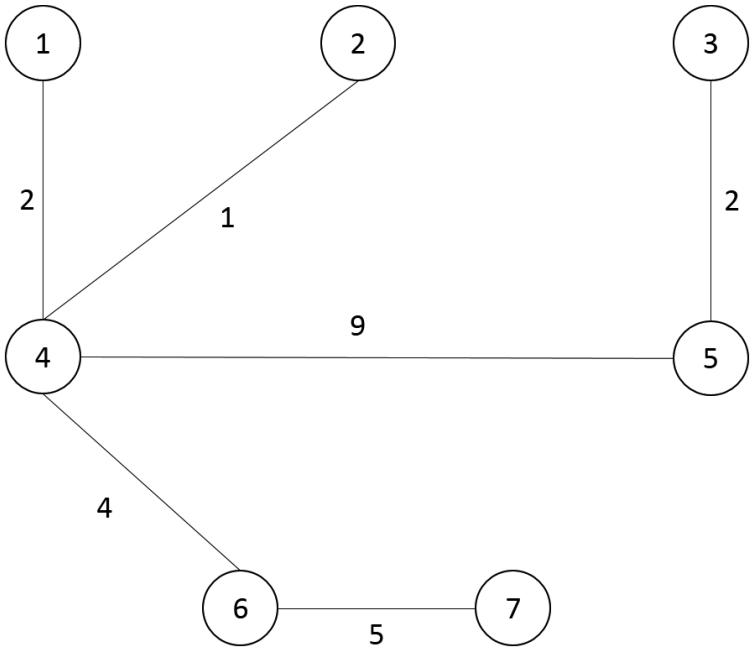
Selezioniamo l'arco a_{67}. Il grafo $T = (N, L \cup \{a_{67}\})$ non ha cicli, allora aggiungiamo l'arco a_{67} a L: $L = \{a_{24}, a_{14}, a_{35}, a_{46}, a_{67}\}$.

$|L|$=5 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 7

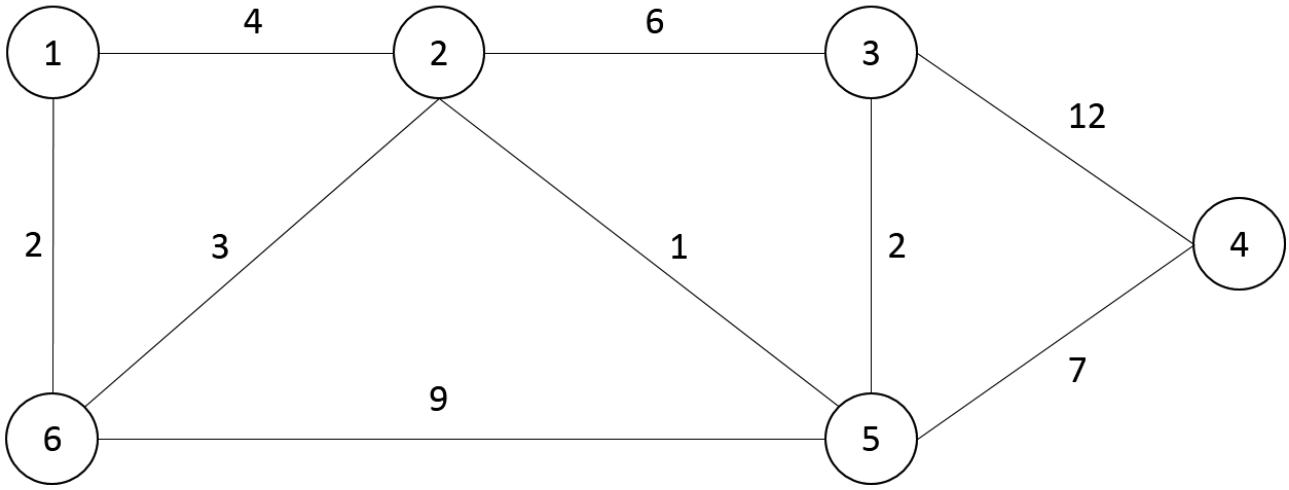
Selezioniamo l'arco a_{45}. Il grafo $T = (N, L \cup \{a_{45}\})$ non ha cicli, allora aggiungiamo l'arco a_{45} a L: $L = \{a_{24}, a_{14}, a_{35}, a_{46}, a_{67}, a_{45}\}$.

$|L|$=6 allora ci fermiamo. L'albero di costo minimo è $T = (N, L)$, rappresentato in figura e di costo 23.


1
2
3
2
1
2
9
4
5
4
6
7
5

Esercizio 2

Sia dato il grafo *G= (N, A)* pesato e non orientato riportato in figura. Applicando l'algoritmo di Kruskal, definire l'albero di costo minimo. Indicare con rigore i vari passi dell'algoritmo.



Soluzione

Passo 0

Inizializziamo l'insieme *L* degli archi dell'albero di costo minimo.

$L = \emptyset$

Inoltre, ordiniamo gli archi del grafo in ordine crescente di costo.

$a_{25} \rightarrow 1$
$a_{16} \rightarrow 2$
$a_{35} \rightarrow 2$
$a_{26} \rightarrow 3$
$a_{12} \rightarrow 4$
$a_{23} \rightarrow 6$
$a_{45} \rightarrow 7$
$a_{56} \rightarrow 9$
$a_{34} \rightarrow 12$

Iniziamo a creare l'albero, aggiungendo gli opportuni archi sulla base della lista appena definita e fermando l'algoritmo quando |L|=|N|-1, ovvero |L|=5.

Passo 1

Selezioniamo l'arco a_{25}. Il grafo $T = (N, L \cup \{a_{25}\})$ non ha cicli, allora aggiungiamo l'arco a_{25} a L: $L = \{a_{25}\}$.

$|L|$=1 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 2

Selezioniamo l'arco a_{16}. Il grafo $T = (N, L \cup \{a_{16}\})$ non ha cicli, allora aggiungiamo l'arco a_{16} a L: $L = \{a_{25}, a_{16}\}$.

$|L|$=2 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 3

Selezioniamo l'arco a_{35}. Il grafo $T = (N, L \cup \{a_{35}\})$ non ha cicli, allora aggiungiamo l'arco a_{35} a L: $L = \{a_{25}, a_{16}, a_{35}\}$.

$|L|$=3 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 4

Selezioniamo l'arco a_{26}. Il grafo $T = (N, L \cup \{a_{26}\})$ non ha cicli, allora aggiungiamo l'arco a_{26} a L: $L = \{a_{25}, a_{16}, a_{35}, a_{26}\}$.

$|L|$=4 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 4

Selezioniamo l'arco a_{12}. Il grafo $T = (N, L \cup \{a_{12}\})$ ha cicli (ad esempio, il cammino 1-2-6-1 è un ciclo), allora non aggiungiamo l'arco a_{12} a L.

$|L|$=4 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 5

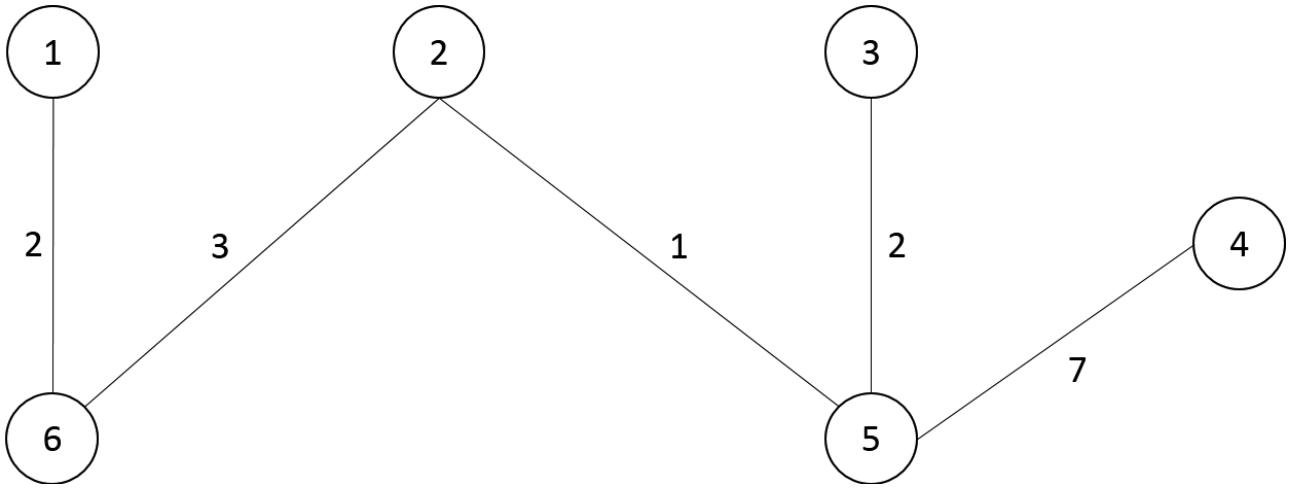
Selezioniamo l'arco a_{23}. Il grafo $T = (N, L \cup \{a_{23}\})$ ha cicli (ad esempio, il cammino 2-3-5-2 è un ciclo), allora non aggiungiamo l'arco a_{23} a L.

$|L|$=4 allora andiamo avanti nell'esecuzione dell'algoritmo.

Passo 6

Selezioniamo l'arco a_{45}. Il grafo $T = (N, L \cup \{a_{45}\})$ non ha cicli, allora aggiungiamo l'arco a_{45} a L: $L = \{a_{25}, a_{16}, a_{35}, a_{26}, a_{45}\}$.

$|L|$=5 allora ci fermiamo. L'albero di costo minimo è $T = (N, L)$, rappresentato in figura.


1
2
3
4
5
6
2
3
1
2
7