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

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

Esercitazione: vari

dr. Manuel Fiorelli

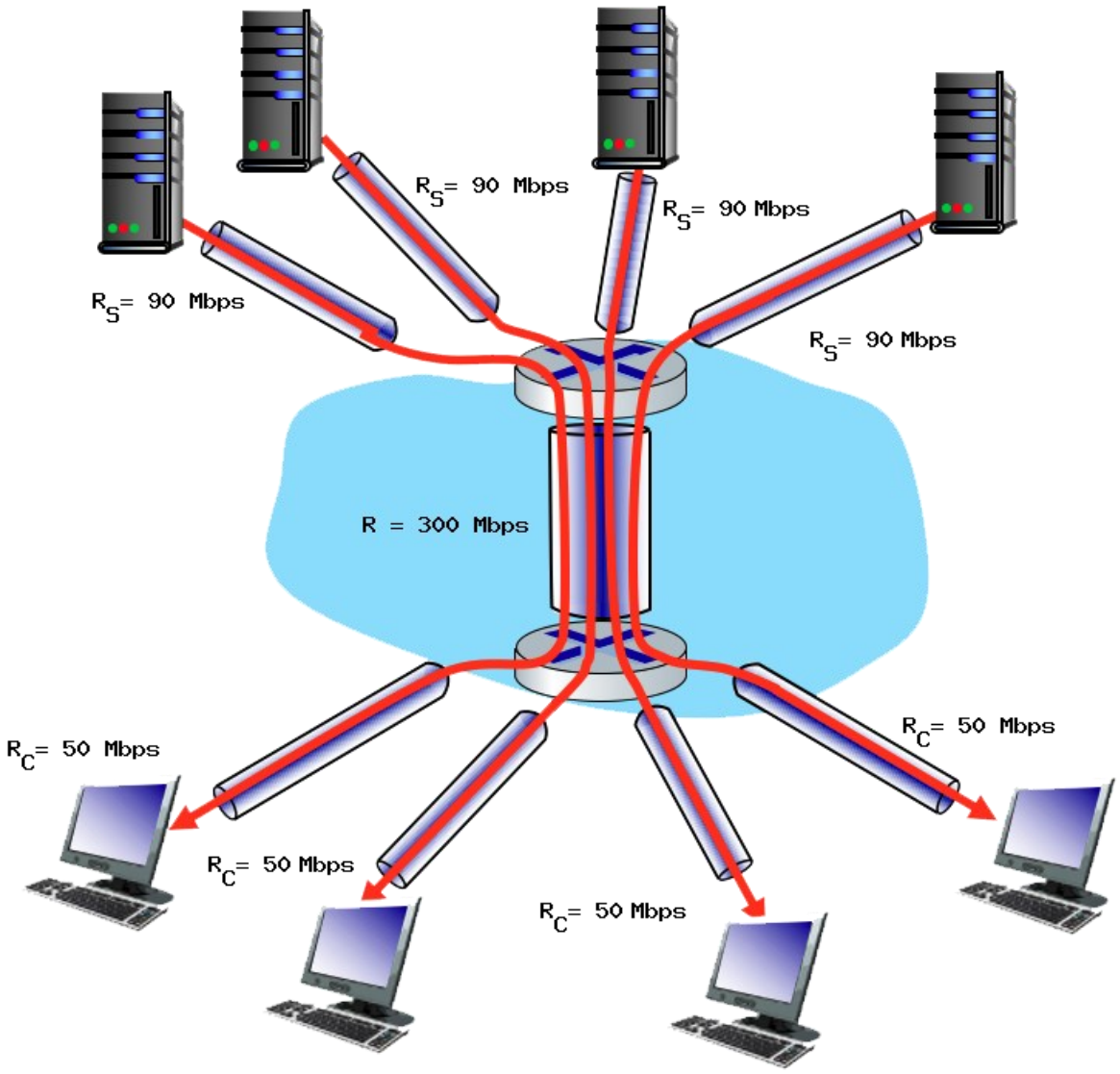
manuel.fiorelli@uniroma2.it

https://art.uniroma2.it/fiorelli

Esercizio 1

Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/end-end-throughput-simple.php

Consider the scenario shown below, with four different servers connected to four different clients over four three-hop paths. The four pairs share a common middle hop with a transmission capacity of R = 300 Mbps. The four links from the servers to the shared link have a transmission capacity of R_S = 90 Mbps. Each of the four links from the shared middle link to a client has a transmission capacity of R_C = 50 Mbps.

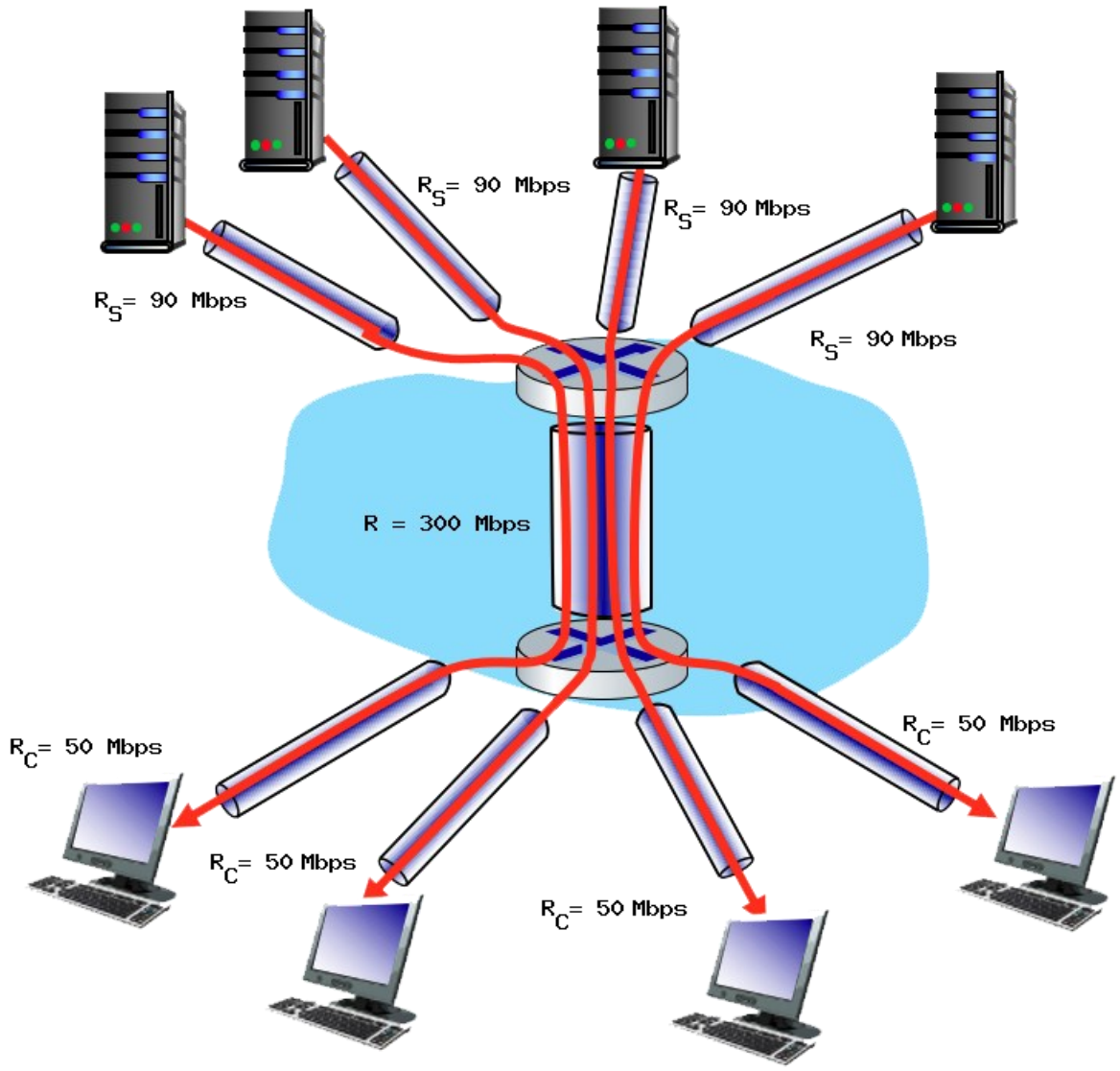


Esercizio 1

Qual è il throughput end-end massimo raggiungibile (in Mbps) per ciascuna delle quattro coppie client-server, supponendo che il collegamento centrale sia condiviso in modo equo (divida equamente la sua velocità di trasmissione)?

$$\min\left\{90, \underbrace{75}_{\frac{300}{4}}, 50\right\} = 50$$

Qual è il collegamento collo di bottiglia (bottleneck)?
R_c perché è quello che limita il throughput end-to-end



Esercizio 1

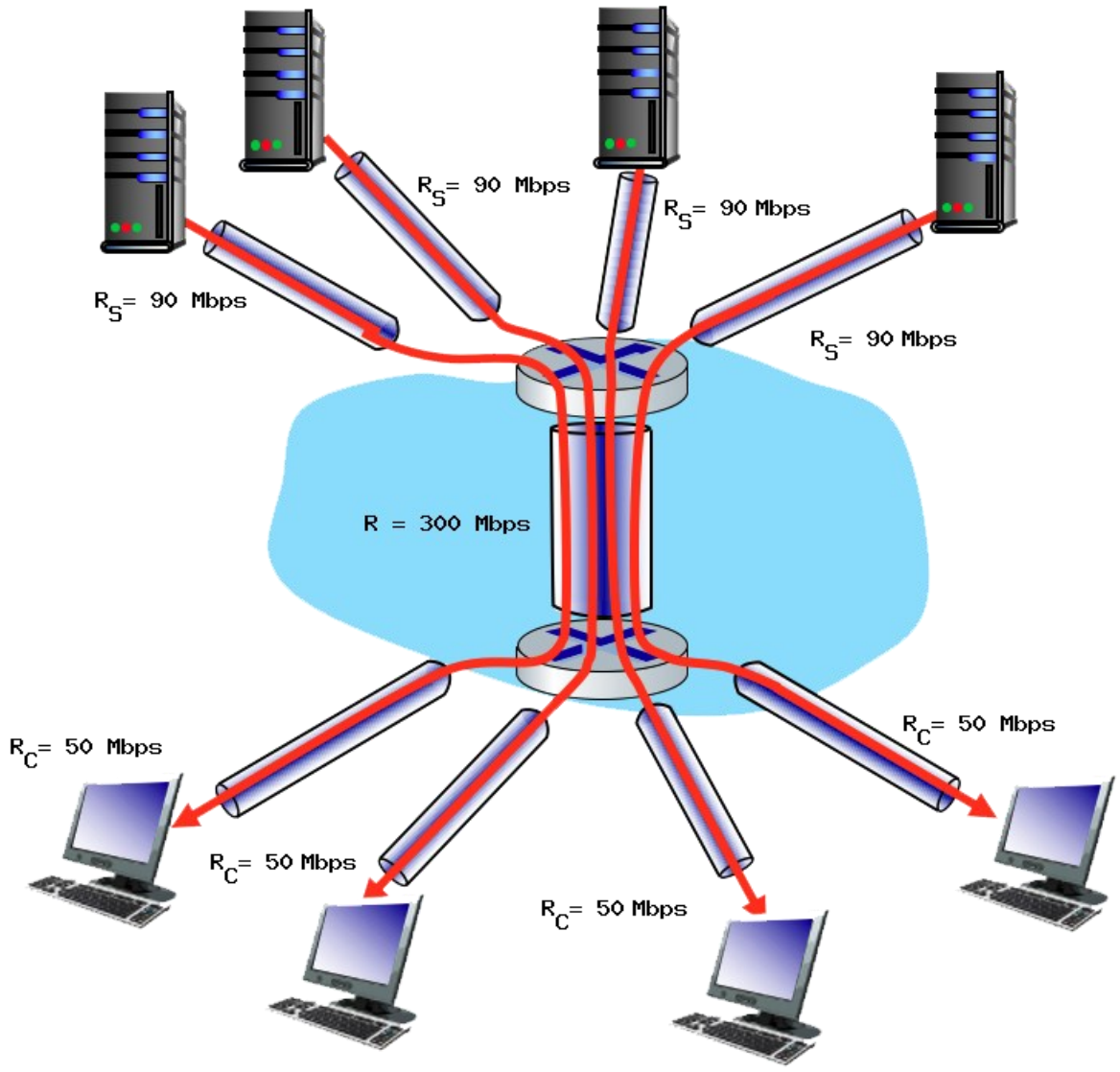
Supponendo che i server stiano inviando alla massima velocità possibile, qual è l'utilizzo (*utilization*) dei collegamenti al server (R_S)?

$$utilization = frazione\ di\ tempo\ in\ cui\ il\ collegamento$$

$$\text{è } attivo = \frac{50}{90} = 0.56$$

da notare che a causa del collo di bottiglia il tasso di invio dei server è inferiore all'ampiezza di banda/velocità di trasmissione del collegamento.

nota: il server trasmette ciascun pacchetto a 90 Mbps (trascurando altri overhead), ma deve inserire delle pause affinché il tasso di invio medio rispetti il vincolo dei 50 Mbps (per effetto della regolazione della finestra di invio operata dal controllo della congestione).



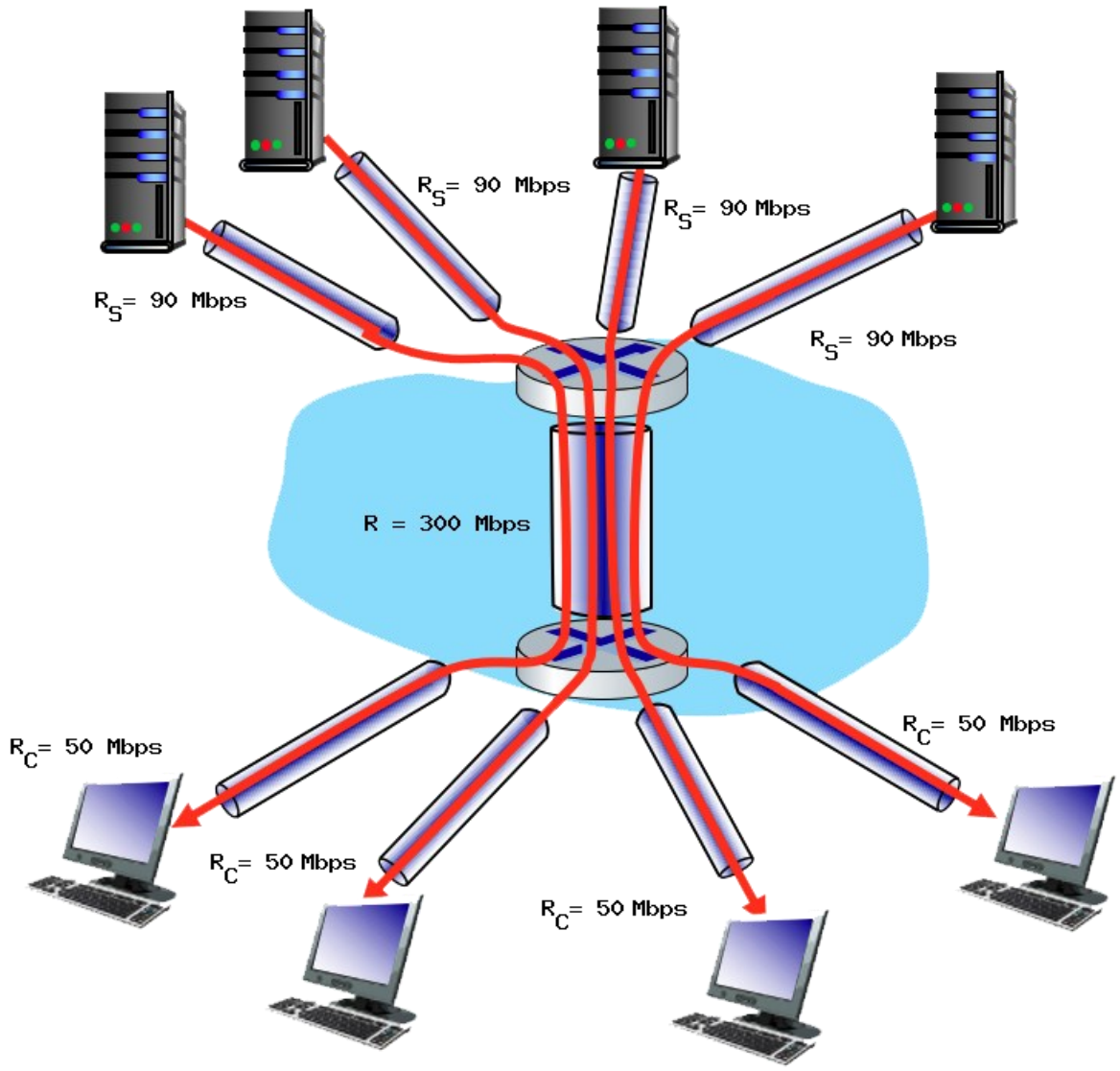
Esercizio 1

Supponendo che i server inviino alla massima velocità possibile, qual è l'utilizzo (utilization) dei collegamenti client (R_C)?

$$\frac{50}{50} = 1.0$$

Supponendo che i server inviino alla massima velocità possibile, qual è l'utilizzo (utilization) del collegamento condiviso (R)?

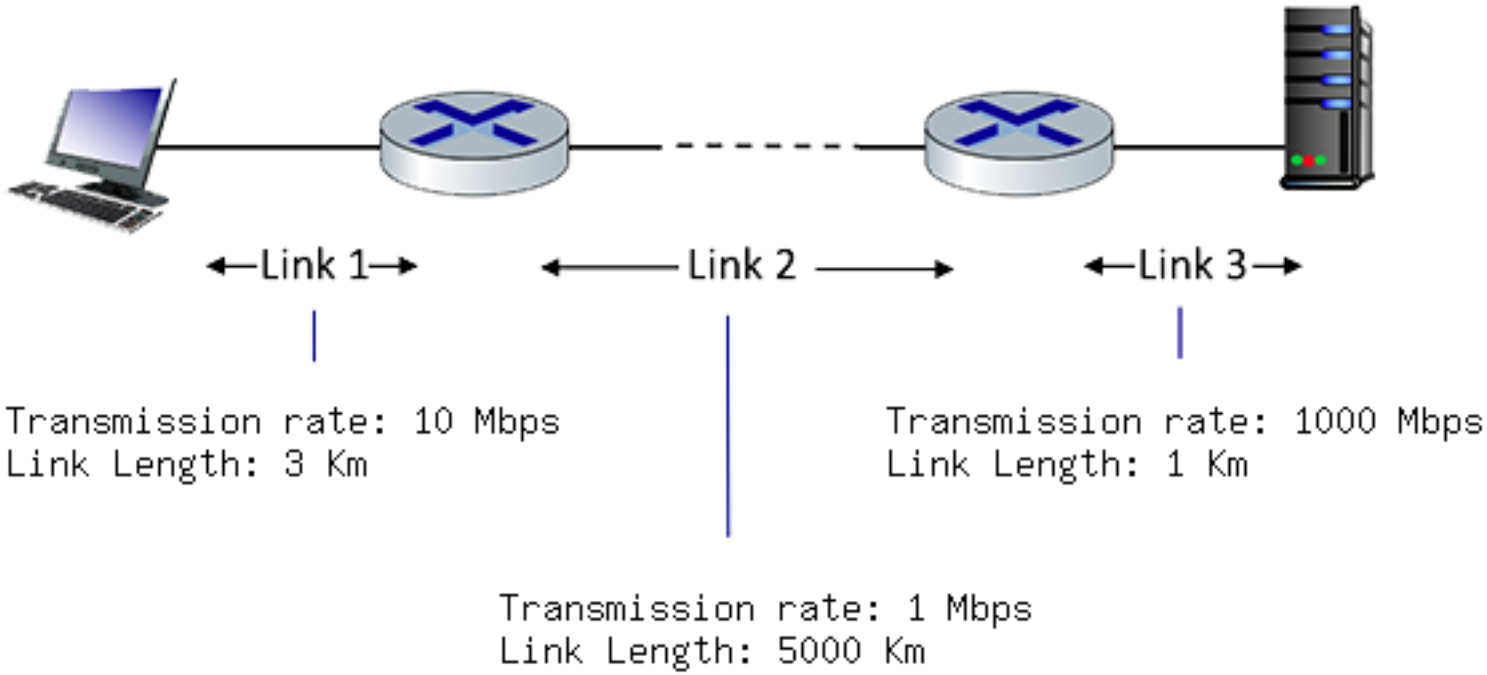
$$\frac{50 \cdot 4}{300} = \frac{200}{300} = 0.67$$



Esercizio 2

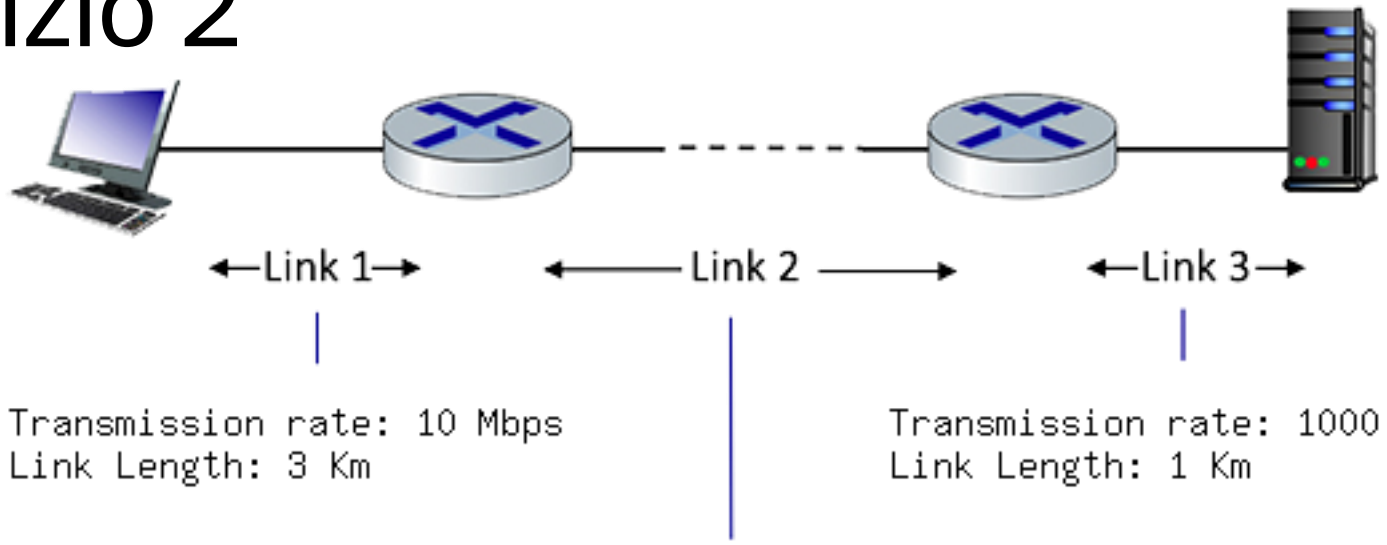
Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/end-end-throughput-simple.php

Consider the figure below, with three links, each with the specified transmission rate and link length.



Assume the length of a packet is 16000 bits. The speed of light propagation delay on each link is 3x10^8 m/sec

Esercizio 2



L = 16000 bit

v = $3 \cdot 10^8$ m/s

ritardo di trasmissione del link 2

$T^1_{trasm} = \frac{L}{R_2} = \frac{16000}{10 \cdot 10^6} = \frac{1.6 \cdot 10^4}{1 \cdot 10^7} = 1.60 \cdot 10^{-3}$ s

ritardo di propagazione del link 1

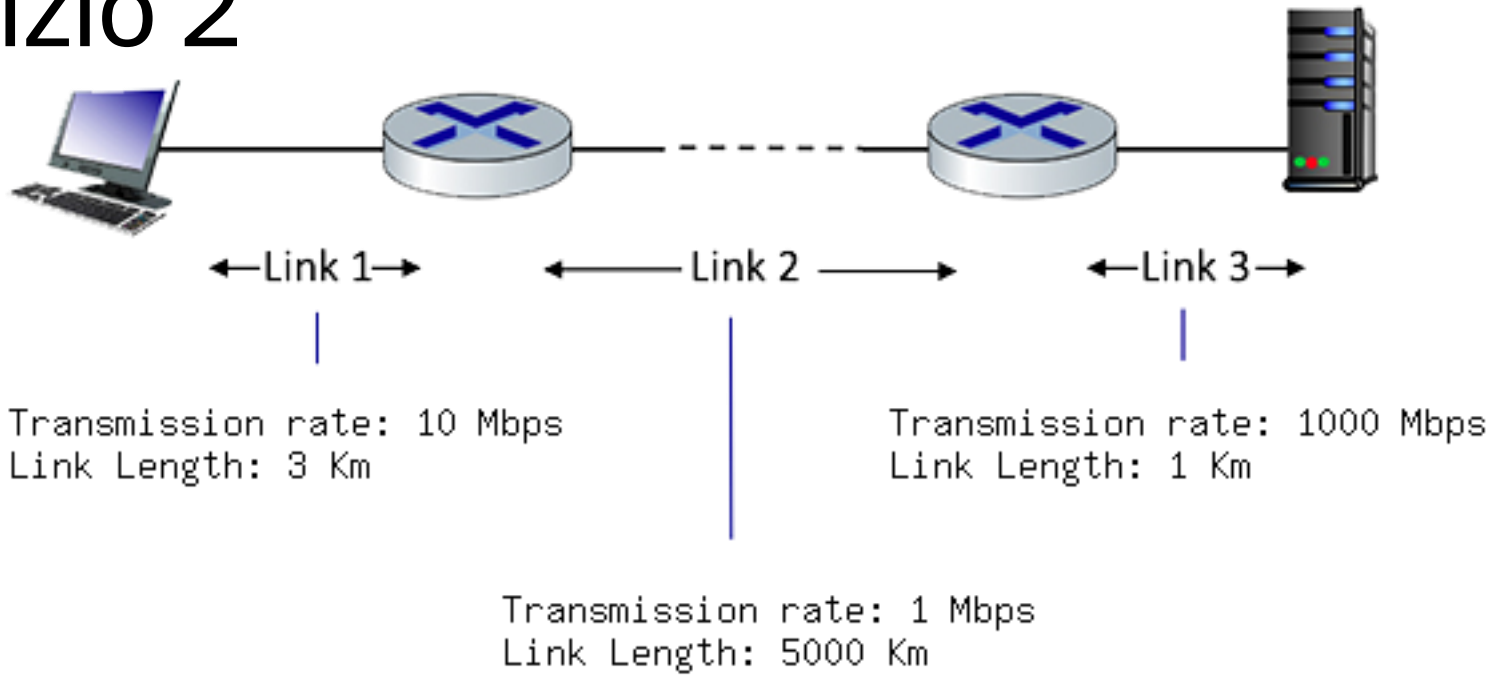
$T^1_{prop} = \frac{d_1}{v} = \frac{3 \cdot 10^3}{3 \cdot 10^8} = 1.00 \cdot 10^{-5}$s

Nota: $10^{-3} > 10^{-5}$ perché $10^{-3} = \frac{1}{10^3} > \frac{1}{10^5} = 10^{-5}$

ritardo totale del link 1

$T^1_{tot} = T^1_{trasm} + T^2_{prop} = 1.60 \cdot 10^{-3} + 1.00 \cdot 10^{-5} = (1.60 + 0.01) \cdot 10^{-3} = 1.61 \cdot 10^{-3}$s

Esercizio 2



L = 16000 bit

v = $3 \cdot 10^8$ m/s

ritardo di trasmissione del link 2

$T^2_{trasm} = \frac{L}{R_2} = \frac{16000}{1 \cdot 10^6} = 1.60 \cdot 10^{-2}$ s

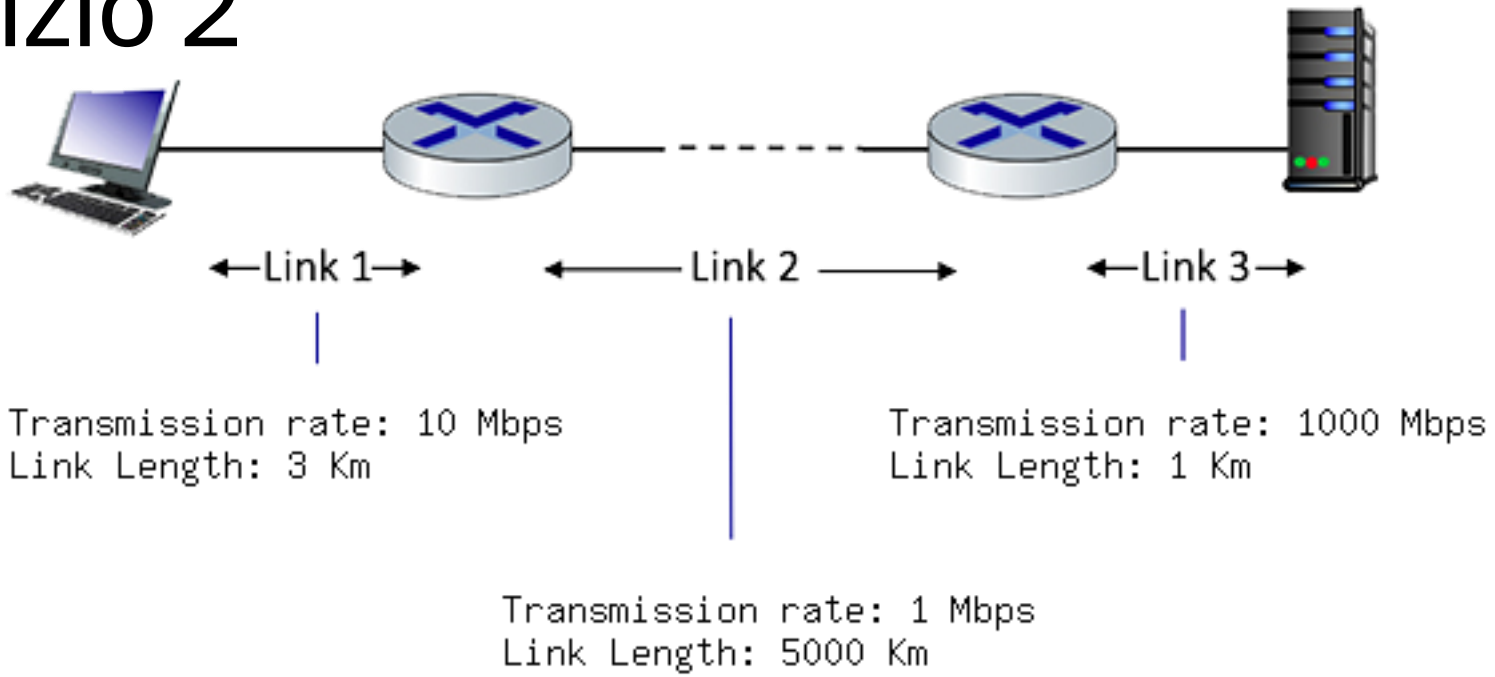
ritardo di propagazione del link 1

$T^2_{prop} = \frac{d_2}{v} = \frac{5 \cdot 10^6}{3 \cdot 10^8} = 1.67 \cdot 10^{-2}$s

ritardo totale del link 2

$T^2_{tot} = T^2_{trasm} + T^2_{prop} = 3.27 \cdot 10^{-2}$s

Esercizio 2



L = 16000 bit

v = $3 \cdot 10^8$ m/s

ritardo di trasmissione del link 3

$T^3_{trasm} = \frac{L}{R_3} = \frac{16000}{1 \cdot 10^9} = 1.60 \cdot 10^{-5}$ s

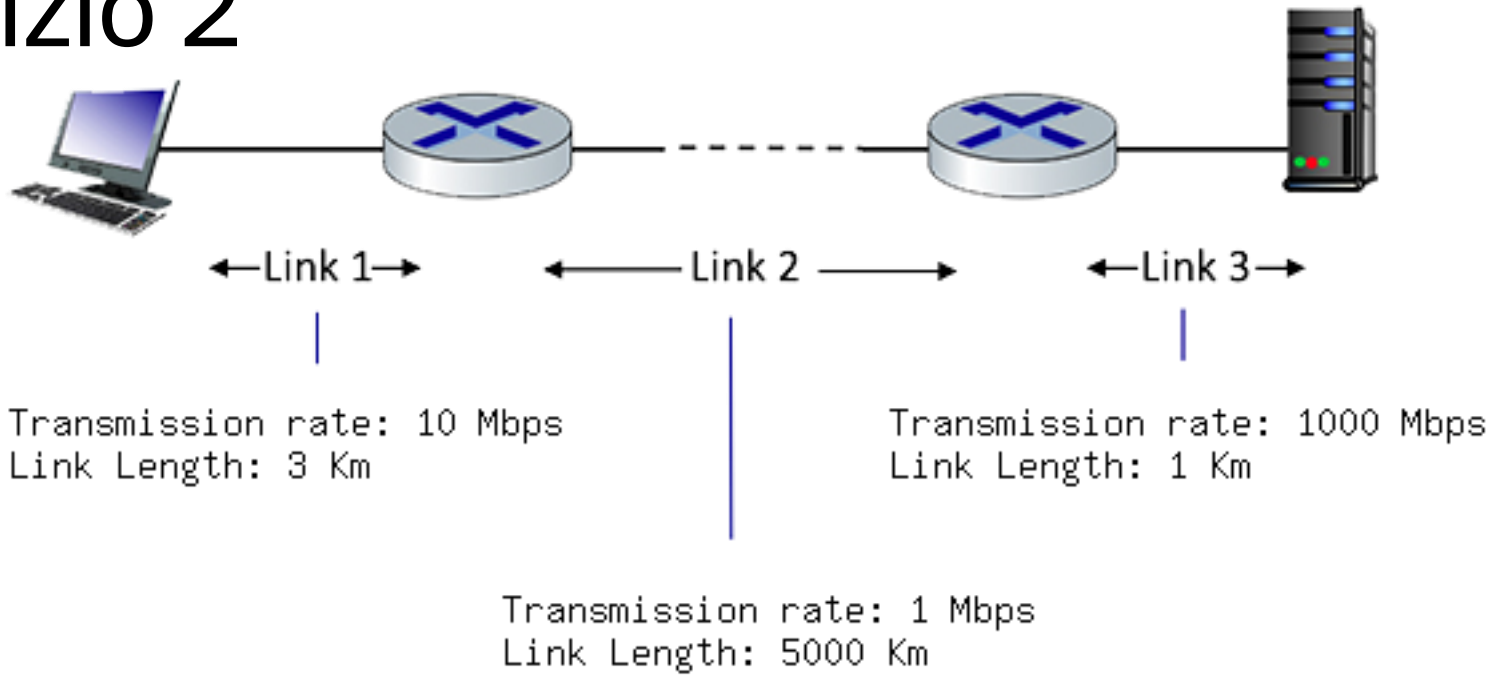
ritardo di propagazione del link 1

$T^3_{prop} = \frac{d_3}{v} = \frac{1 \cdot 10^3}{3 \cdot 10^8} = 3.33 \cdot 10^{-6}$s

ritardo totale del link 3

$T^3_{tot} = T^3_{trasm} + T^3_{prop} = 1.93 \cdot 10^{-5}$s

Esercizio 2



L = 16000 bit

v = $3 \cdot 10^8$ m/s

ritardo totale

$T_{tot} = T_{tot}^1 + T_{tot}^2 + T_{tot}^3 = 1.61 \cdot 10^{-3} + 3.27 \cdot 10^{-2} + 1.93 \cdot 10^{-5} = 0.161 \cdot 10^{-2} + 3.27 \cdot 10^{-2} + 0.00193 \cdot 10^{-2} = 3.43 \cdot 10^{-2}$s

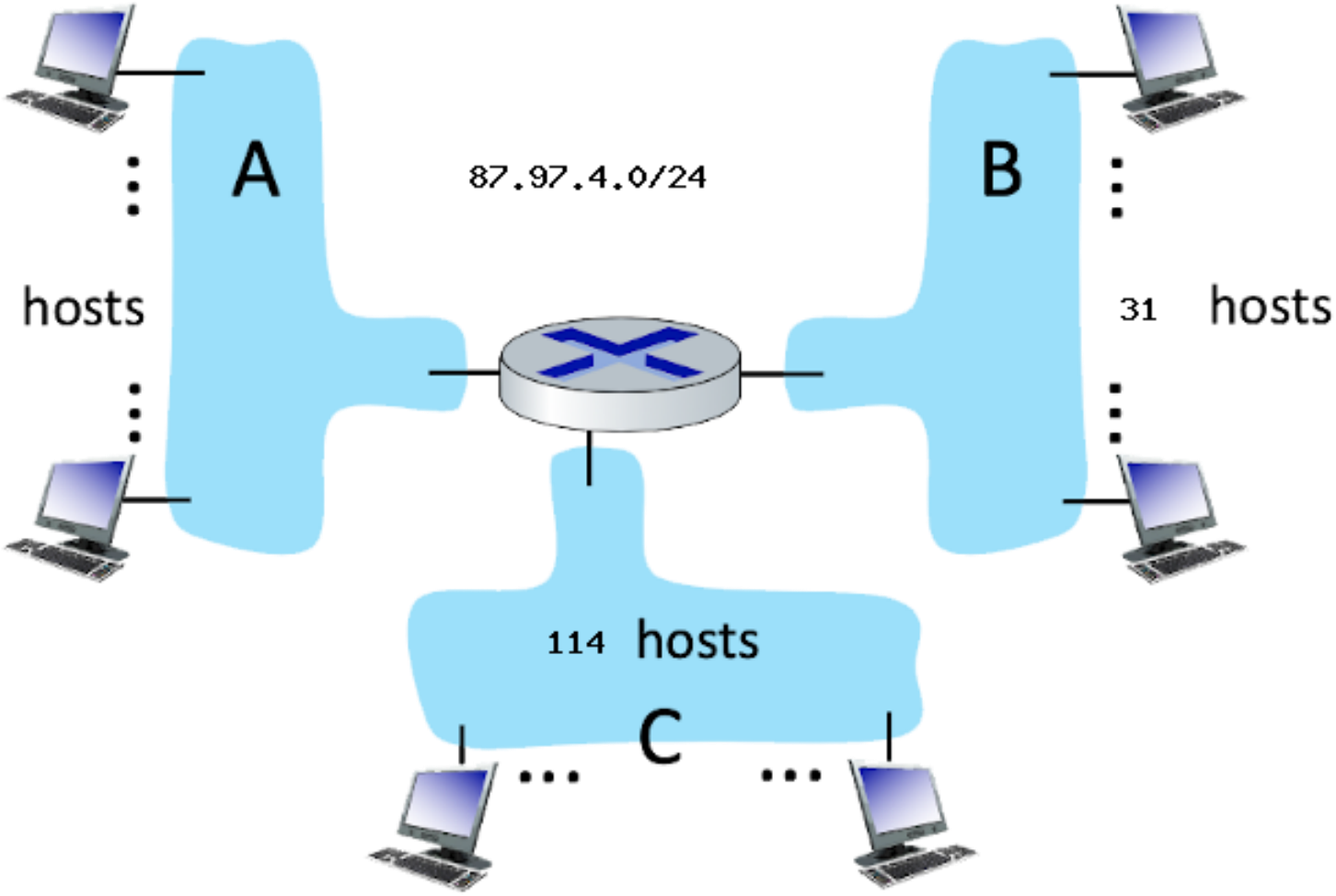
nota: il fatto di poter sommare i ritardi si riferisce al **singolo pacchetto** come conseguenza del comportamento store-and-forward dei commutatori di pacchetto. Per confronto, vedere gli esercizi passati in cui abbiamo calcolato il tempo totale richiesto per trasferire più pacchetti.

Esercizio 3

Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/subnet_addressing.php

Consider the router and the three attached subnets below (A, B, and C). The number of hosts is also shown below. The subnets share the 24 high-order bits of the address space: 87.97.4.0/24

Assign subnet addresses to each of the subnets (A, B, and C) so that the amount of address space assigned is minimal, and at the same time leaving the largest possible contiguous address space available for assignment if a new subnet were to be added.



Esercizio 3

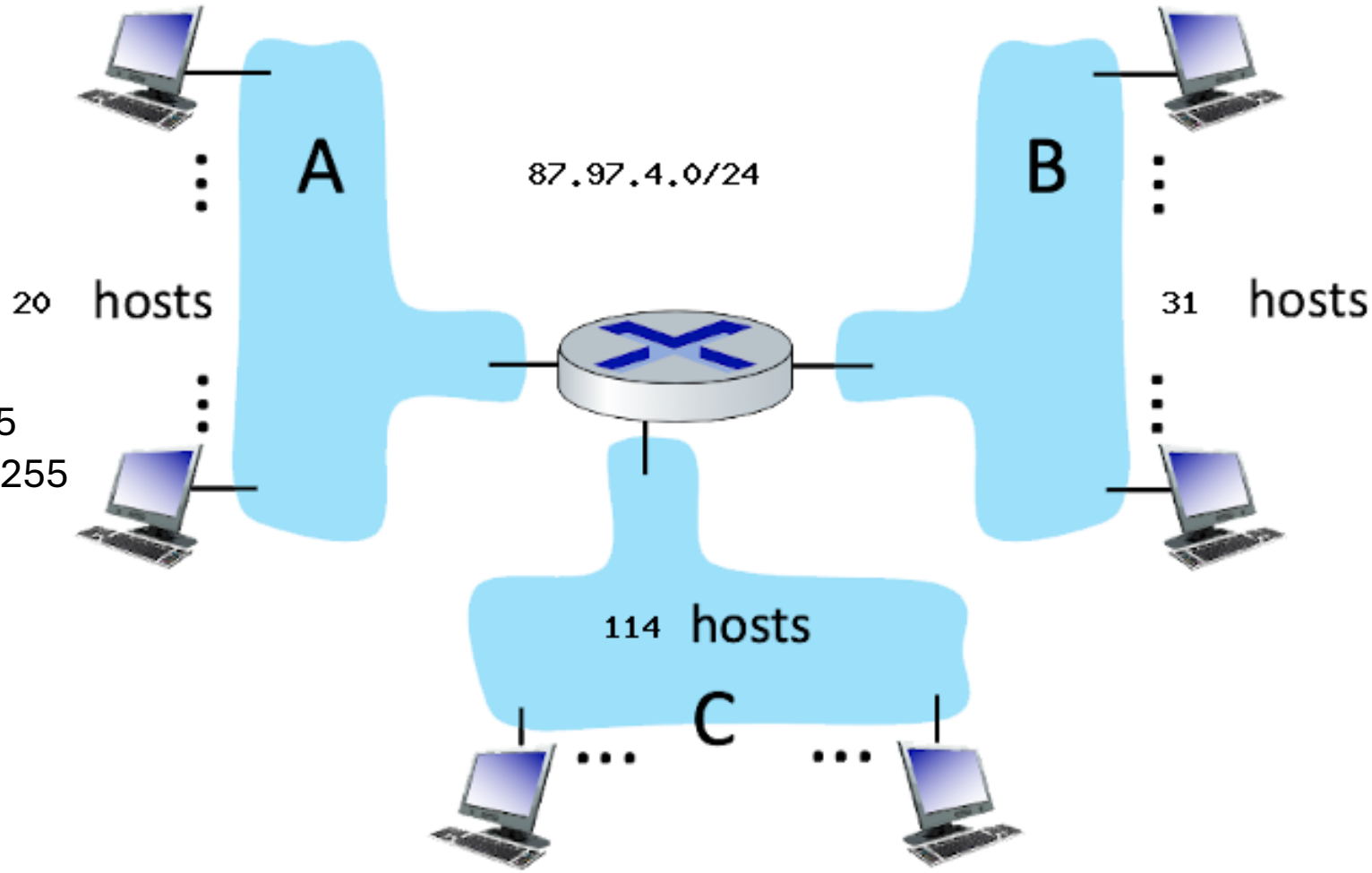
Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/subnet_addressing.php

Lo spazio degli indirizzi è pubblico o privato?

Blocchi di indirizzi IPv4 privati

10.0.0.0/8	10.0.0.0 – 10.255.255.255
172.16.0.0/12	172.16.0.0 – 172.31.255.255
192.168.0.0/16	192.168.0.0 – 192.168.255.255

87.97.4.0 non ricade in nessuno di questi intervalli (dettagliare il motivo) quindi è pubblico!

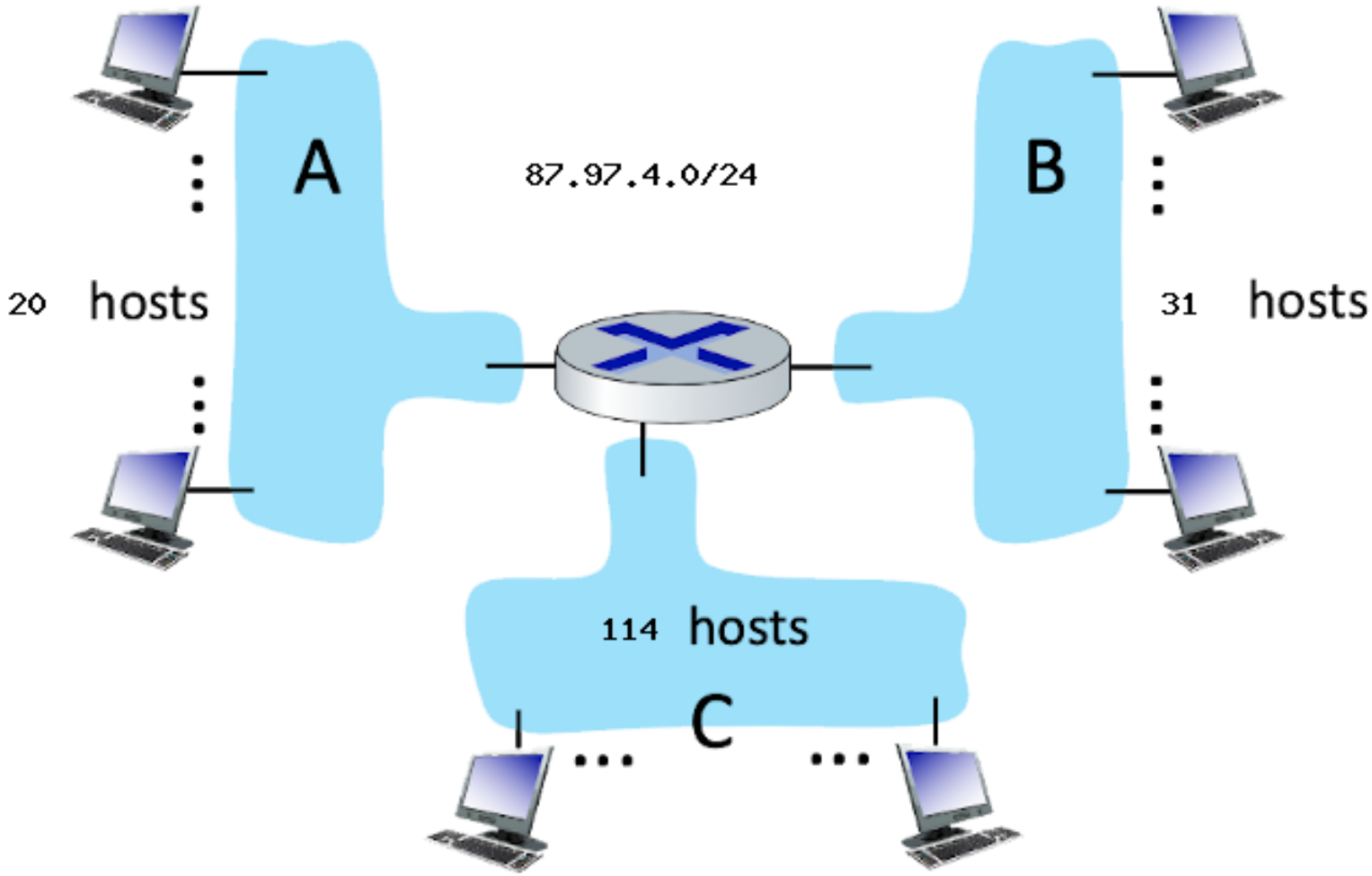


Esercizio 3

Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/subnet_addressing.php

Quanti host possono essere presenti in questo spazio di indirizzi?

La parte di host è $32 - 24 = 8$, pertanto possono esserci $2^8 - 2 = 254$ host.

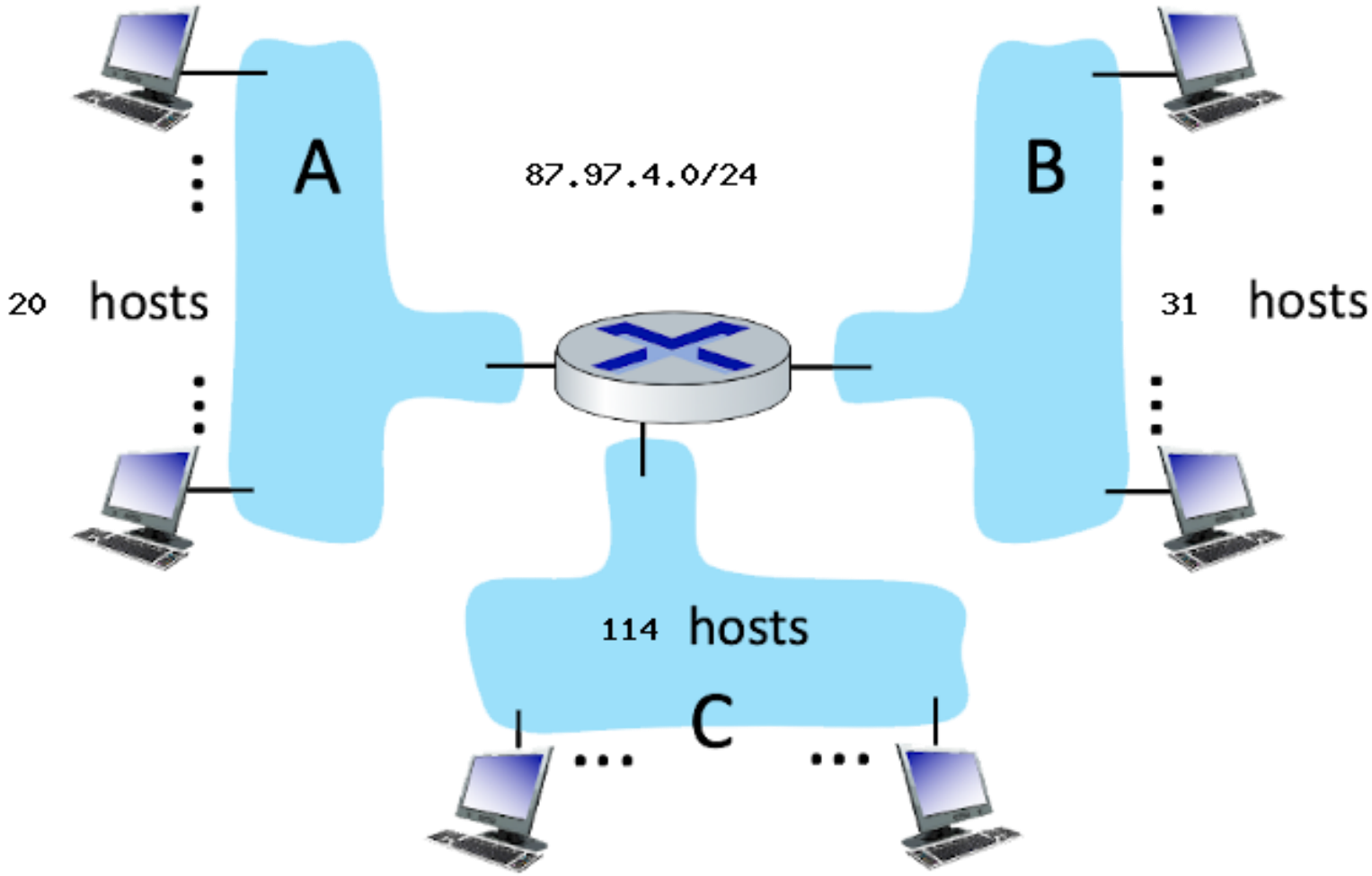


Esercizio 3

Sottorete	Interfacce
A	21
B	32
C	114

ordino ↓

Sottorete	Interfacce
C	114
B	32
A	21

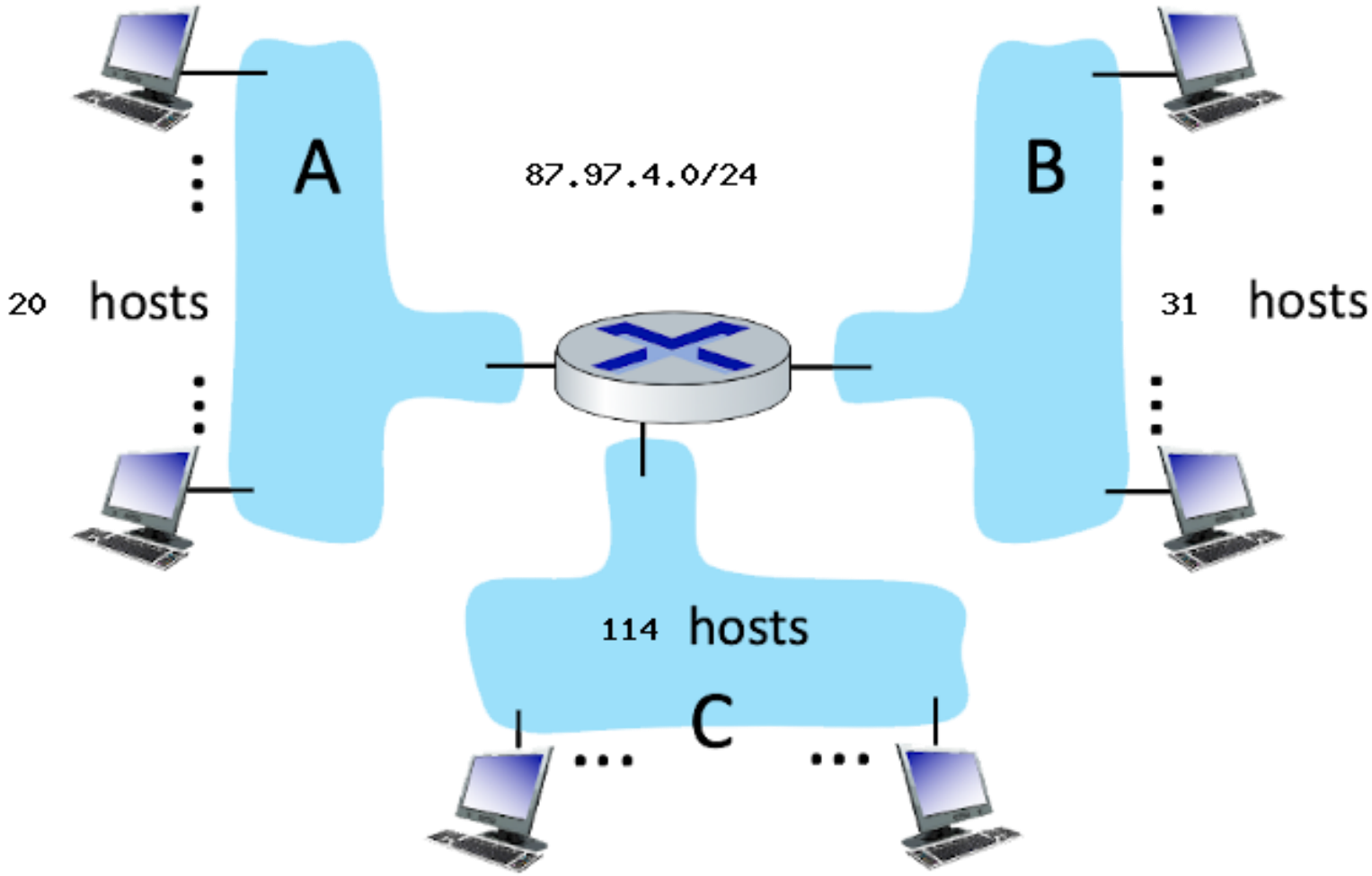


Esercizio 3

Sottorete	Interfacce
C	114
B	32
A	21

Trovo il più grande n tale che
$2^{32-n} - 2 \geq interfacce$

Sottorete	Prefisso
C	/25
B	/26
A	/27



Esercizio 3

Sottorete	Prefisso
C	/25
B	/26
A	/27

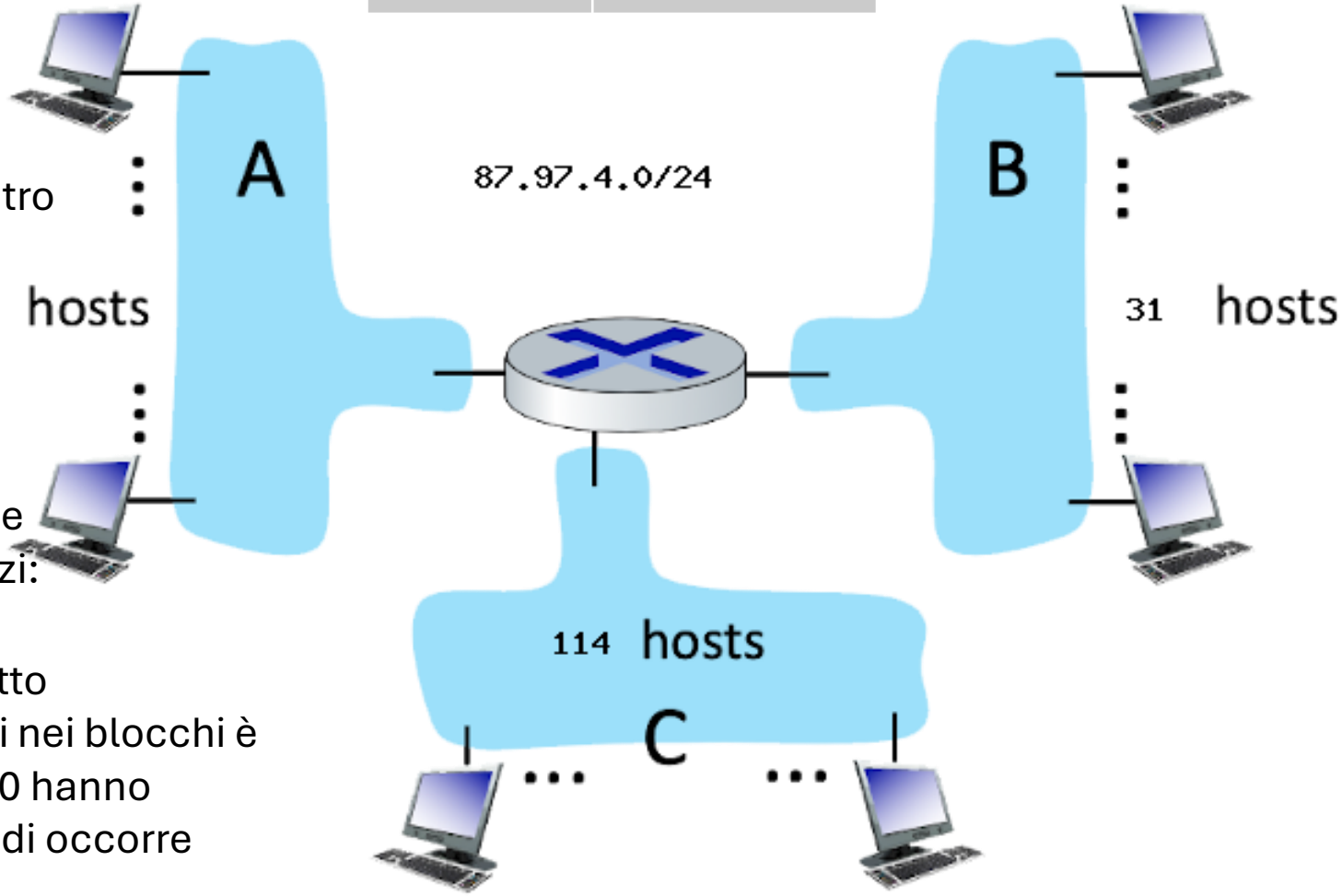
87.97.4.0/24

L'indirizzo del primo blocco non cambia perché ho portato dentro solo bit 0

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25

Ho usato la tecnica per trovare il blocco successivo sommando all'indirizzo di rete del precedente il numero totale di indirizzi: $128_{10} = 10000000_2$ per blocchi /25

Nota: Si deve incrementare un byte/ottetto precedente se il numero totale di indirizzi nei blocchi è un numero con più di 8 bit: es. blocchi /20 hanno dimensione $2^{12} = 10000|00000000_2$, quindi occorre sommare 16 sul terzo byte (da sinistra).



Esercizio 3

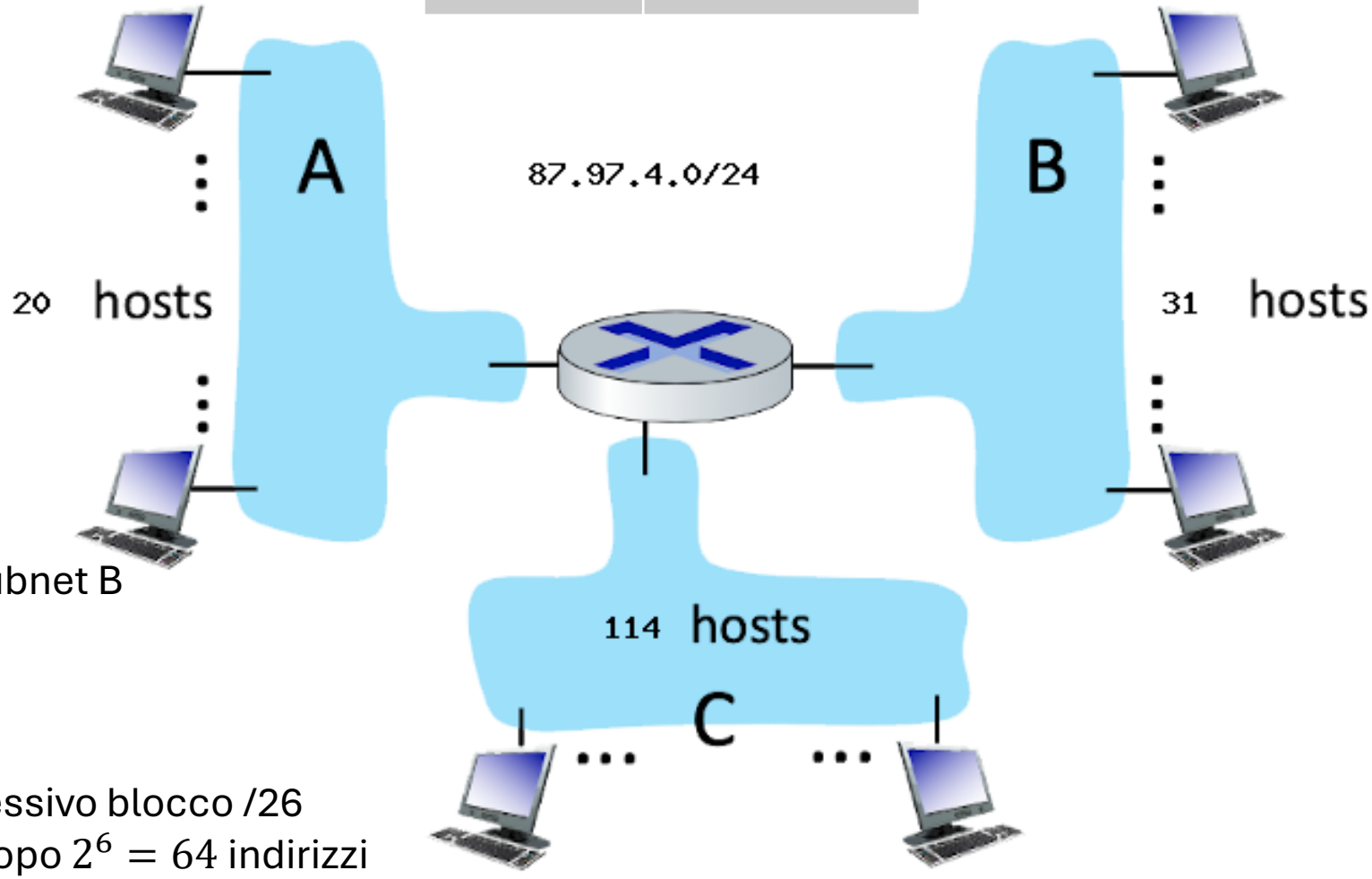
Sottorete	Prefisso
C	/25
B	/26
A	/27

87.97.4.0/24

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25
  - 87.97.4.128/26 → Subnet B
  - 87.97.4.192/26

L'indirizzo del primo blocco non cambia perché ho portato dentro solo bit 0

Il successivo blocco /26 viene dopo $2^6 = 64$ indirizzi



Esercizio 3

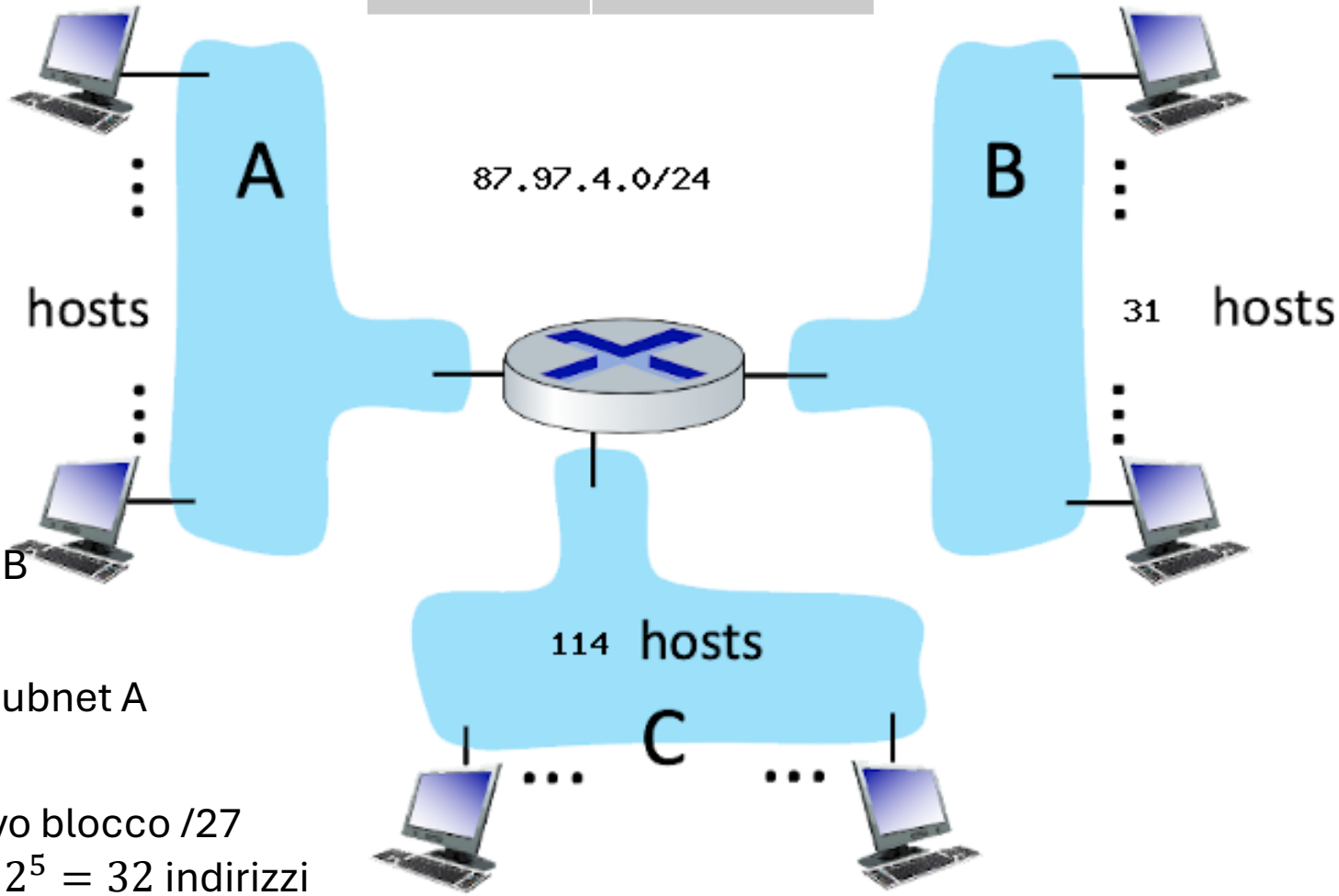
Sottorete	Prefisso
C	/25
B	/26
A	/27

87.97.4.0/24

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25
  - 87.97.4.128/26 → Subnet B
  - 87.97.4.192/26
    - 87.97.4.192/27 → Subnet A
    - 87.97.4.224/27

L'indirizzo del primo blocco non cambia perché ho portato dentro solo bit 0

Il successivo blocco /27 viene dopo $2^5 = 32$ indirizzi

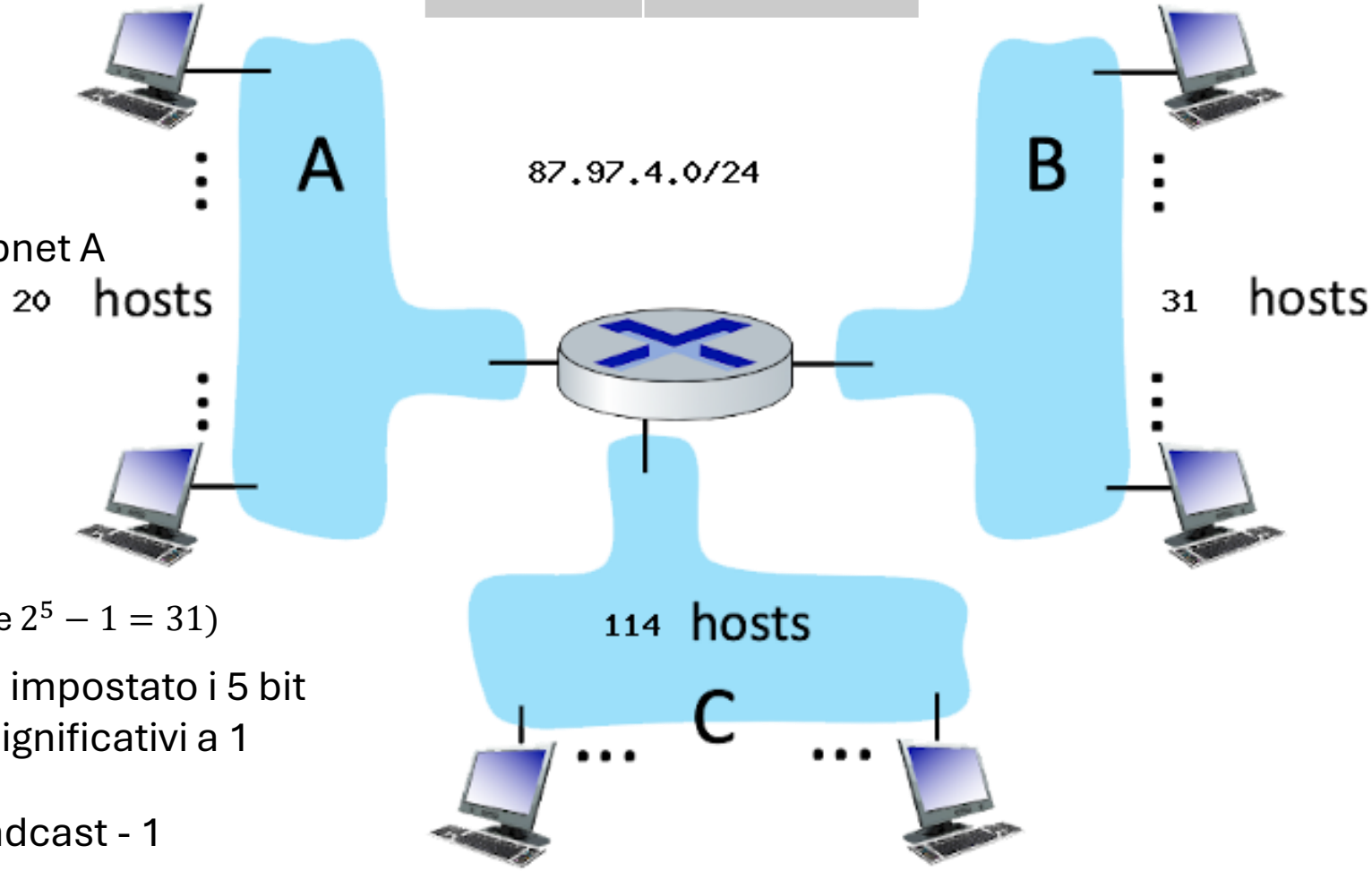


Esercizio 3

87.97.4.0/24

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25
  - 87.97.4.128/26 → Subnet B
  - 87.97.4.192/26
    - 87.97.4.192/27 → Subnet A
    - 87.97.4.224/27

Sottorete	Prefisso
C	/25
B	/26
A	/27



Subnet A
prefisso 87.97.4.192/27
broadcast 87.97.4.223 (occorre sommare $2^5 - 1 = 31$)
Intervallo assegnabile:
87.97.4.193 - 87.97.4.222

avendo impostato i 5 bit meno significativi a 1

Indirizzo di rete + 1

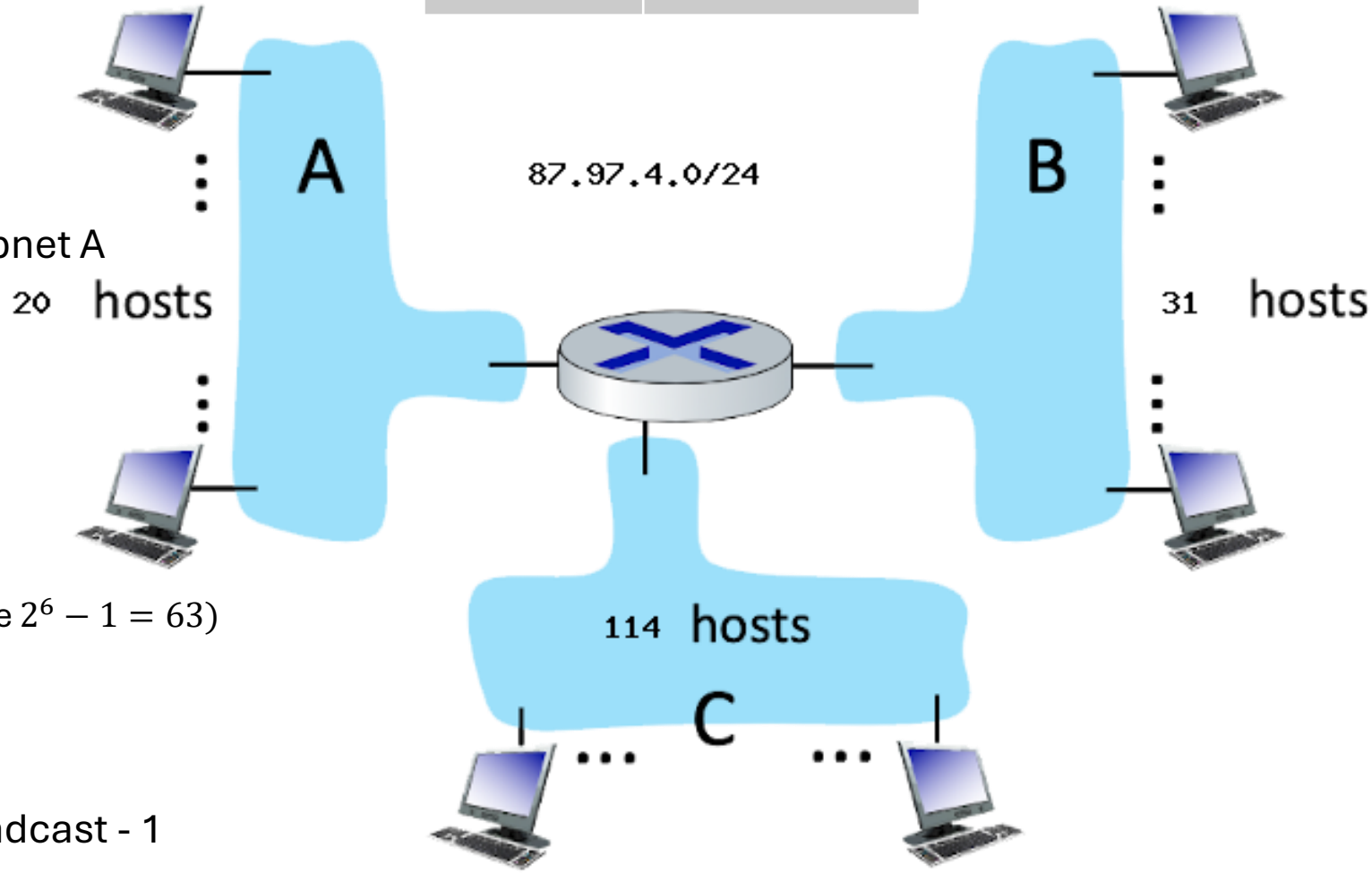
Indirizzo di broadcast - 1

Esercizio 3

87.97.4.0/24

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25
  - 87.97.4.128/26 → Subnet B
  - 87.97.4.192/26
    - 87.97.4.192/27 → Subnet A
    - 87.97.4.224/27

Sottorete	Prefisso
C	/25
B	/26
A	/27



Subnet B
prefisso 87.97.4.128/26
broadcast 87.97.4.191 (occorre sommare $2^6 - 1 = 63$)
Intervallo assegnabile:
87.97.4.129 - 87.97.4.190

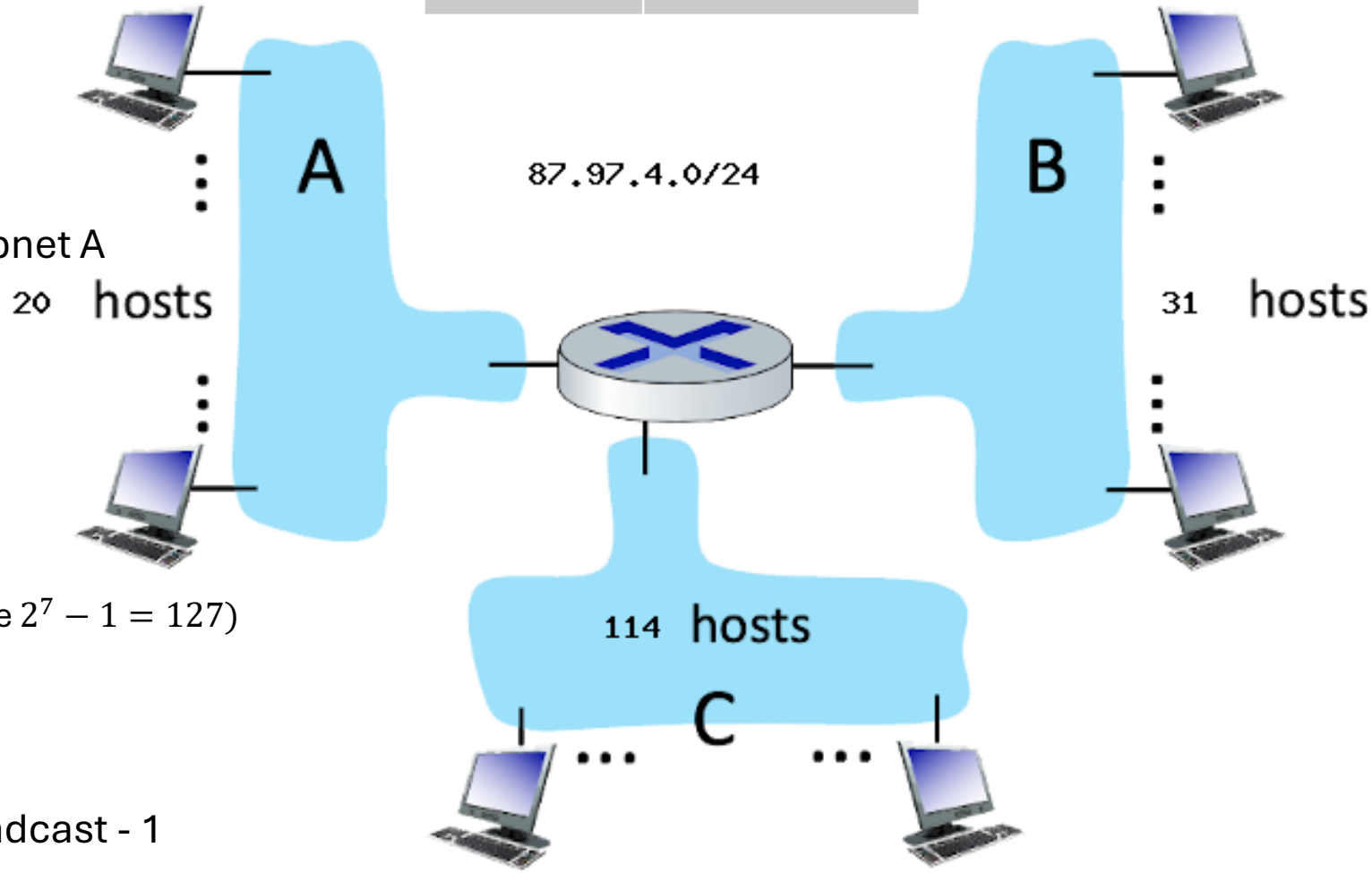
Indirizzo di rete + 1 Indirizzo di broadcast - 1

Esercizio 3

87.97.4.0/24

- 87.97.4.0/25 → Subnet C
- 87.97.4.128/25
  - 87.97.4.128/26 → Subnet B
  - 87.97.4.192/26
    - 87.97.4.192/27 → Subnet A
    - 87.97.4.224/27

Sottorete	Prefisso
C	/25
B	/26
A	/27



Subnet C
prefisso 87.97.4.0/25
broadcast 87.97.4.127 (occorre sommare $2^7 - 1 = 127$)
Intervallo assegnabile:
87.97.4.1 - 87.97.4.126

Indirizzo di rete + 1 Indirizzo di broadcast - 1

Esercizio 4 (vedi esercizio originale in Cap 3, problema P49)

Si consideri il caso in cui una **singola connessione TCP Reno** (pertanto il controllo di congestione è basato sulle perdite) attraversa due collegamenti, di capacità C' e C tale che C' > C, come in figura.

Dimostrare che se la dimensione *B* del buffer davanti al secondo collegamento è tale che

$$B \geq C * (2 * ritardo\ di\ propagazione\ tra\ mittente\ e\ destinatario)$$

allora il collegamento collo di bottiglia sarà sempre occupato a trasmettere.

Esercizio 4 (soluzione)

Assumiamo per semplicità che:

- TCP invii segmenti (interi, di lunghezza 1 MSS ciascuno) in cicli (round) di trasmissione, di durata pari a RTT, incrementando di 1 MSS (quindi di un segmento) la dimensione della finestra di congestione all'inizio del ciclo (round) successivo
- C sia espresso in pacchetti (segmenti) al secondo

Assumendo che C' > C, il mittente continuerà a aumentare l'ampiezza della finestra e quindi la velocità di trasmissione, fino a eccedere la capacità del collegamento collo di bottiglia, riempendo il suo buffer

-> *takeaway*: per quanto possano essere grandi i buffer sul collegamento collo di bottiglia, TCP li riempirà causando una perdita!

Esercizio 4 (soluzione)

Sia W la dimensione della finestra quando viene rilevata la perdita (quindi la dimensione massima della finestra), TCP dimezza la finestra a W/2 (ignoriamo slow start e fast recovery!).

Siccome ci sono W segmenti in transito non ancora riscontrati, TCP non può trasmettere segmenti finché non riceve W/2 ACK.

Se il collegamento collo di bottiglia non è mai a corto di pacchetti, vuol dire che trasmette a un tasso costante C, quindi i riscontri arrivano al mittente con la stessa frequenza C, che dunque dovrà attendere per $\frac{\frac{W}{2}}{C}$ cioè $\frac{W}{2C}$.

Durante questo periodo il collegamento avrà prelevato dal buffer $\frac{W}{2}$ pacchetti

pertanto il buffer deve essere $B \geq \frac{W}{2}$

Esercizio 4 (soluzione)

Quando il mittente riprende a trasmettere, il buffer è vuoto, quindi per evitare che il collegamento collo di bottiglia sia inutilizzato occorre che il tasso di invio del mittente $\frac{\frac{W}{2}}{RTT} = \frac{W}{2RTT}$ sia uguale a C.

In altre parole,

$$\frac{W}{2RTT} = C \rightarrow W = 2 \cdot RTT \cdot C$$

Sostituendo questa espressione nella disuguaglianza di prima, otteniamo:

$$B \geq RTT \cdot C$$

Quando il mittente riprende a trasmettere il buffer è vuoto, quindi non c'è ritardo di accodamento e trascurando le altre componenti del ritardo (si ripassi quali sono!) possiamo approssimare RTT come il doppio del ritardo di propagazione tra mittente e destinatario, dimostrando come richiesto che:

$$B \geq C * (2 * ritardo\ di\ propagazione\ tra\ mittente\ e\ destinatario)$$

Esercizio 4 (soluzione)

Si noti che:

- Se il buffer fosse più piccolo, il collegamento andrebbe a corto di pacchetti durante la pausa del mittente. Ne seguirebbe che il collegamento non potrebbe essere occupato tutto il tempo a trasmettere pacchetti, con una conseguente perdita di throughput
- Se il buffer fosse più grande, il buffer non si svuoterebbe mai (cosa che in realtà abbiamo assunto nella dimostrazione), determinando la presenza di un numero minimo di pacchetti nel buffer e quindi di un ritardo di accordamento minimo non nullo → peggioramento dell'RTT → problema per applicazioni real-time
- Alla fine, la scelta migliore è quella di dimensionare il buffer uguale alla dimensione minima trovata

Esercizio 4 (soluzione)

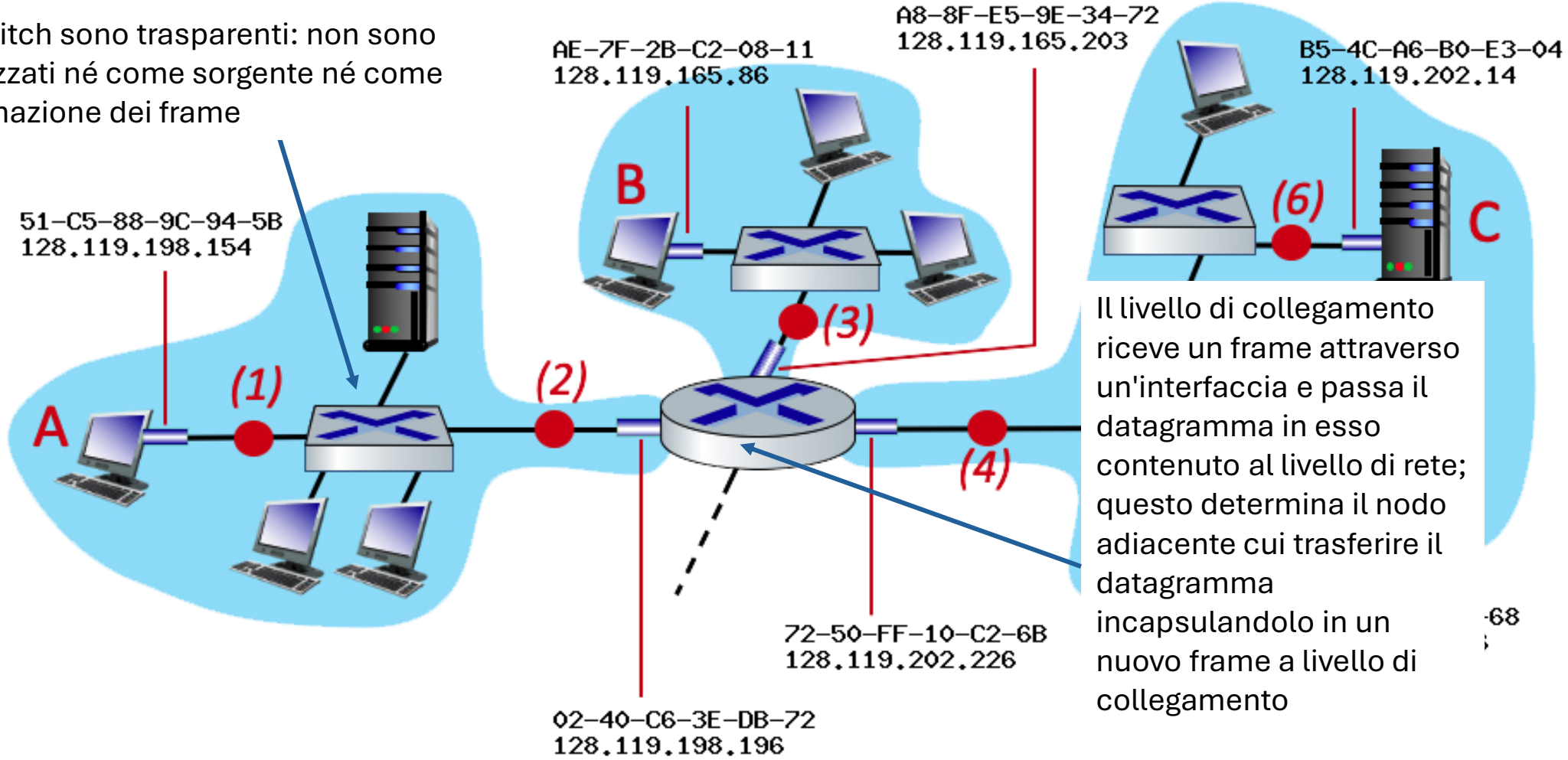
Note:

- Questo esercizio voleva dare un'idea della ragione della vecchia *rule of thumb* per il dimensionamento dei buffer dei router. Si noti che questa menziona in realtà l'RTT medio!!!!!
- Si ricordi che è stata recentemente proposta una nuova regola con a denominatore $\sqrt{N}$ dove *N* è il numero di flussi TCP indipendenti che attraversano un collegamento (si veda nelle slide e nel libro le altre specifiche assunzioni sotto cui vale): questa situazione si verifica nei collegamenti di *backbone* attraversati da molteplici flussi indipendenti, non sincronizzati tra loro

Esercizio 5

Esercizio interattivo del libro:
https://gaia.cs.umass.edu/kurose_ross/interactive/link_layer_addressing.php

gli switch sono trasparenti: non sono indirizzati né come sorgente né come destinazione dei frame



Il livello di collegamento riceve un frame attraverso un'interfaccia e passa il datagramma in esso contenuto al livello di rete; questo determina il nodo adiacente cui trasferire il datagramma incapsulandolo in un nuovo frame a livello di collegamento