Reti di Calcolatori I

Prof. Roberto Canonico
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Corso di Laurea in Ingegneria Informatica

A.A. 2019-2020

Il livello trasporto:

controllo di congestione in TCP

I lucidi presentati al corso sono uno strumento didattico che NON sostituisce i testi indicati nel programma del corso

Università degli Studi di Napoli Federico II

Nota di copyright per le slide COMICS

Nota di Copyright

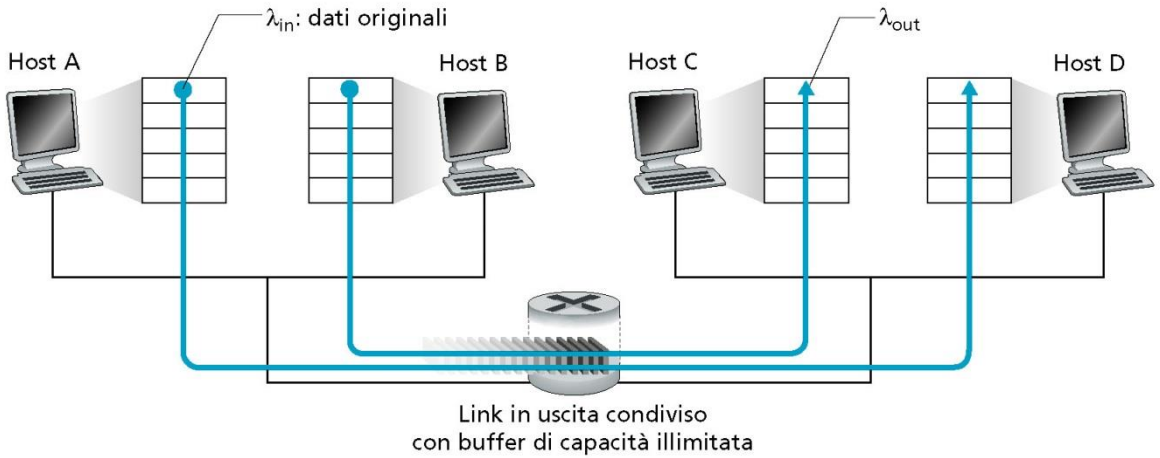
Autori:
Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,
Marcello Esposito, Roberto Canonico, Giorgio Ventre

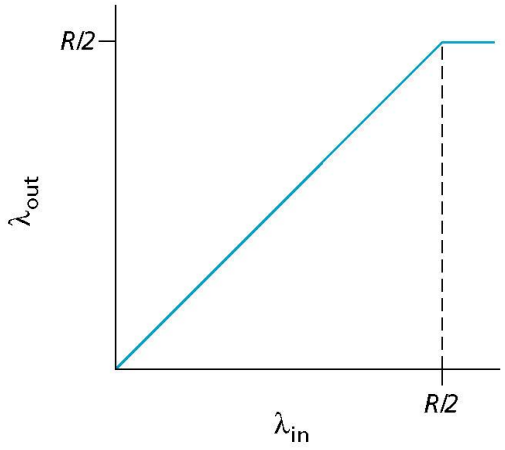
Controllo della congestione

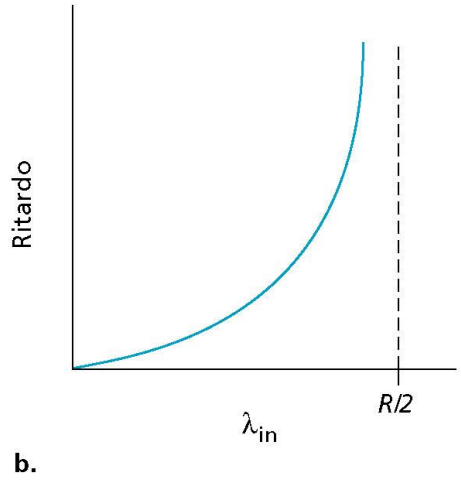
- Congestione nella rete
  - Tecnicamente dovuta a:
    - un numero elevato di sorgenti di traffico
    - sorgenti di traffico che inviano troppi dati
    - traffico inviato ad una frequenza troppo elevata
  - In presenza di questi fenomeni, singoli o concomitanti, la rete è sovraccarica
    - Effetti:
      - perdita di pacchetti:
        - » buffer overflow nei router
      - ritardi nell'inoltro dei pacchetti:
        - » accodamenti nei buffer dei router
      - scarso utilizzo delle risorse di rete

Effetti della congestione: esempi (1/2)

- 2 mittenti
- 2 riceventi
- 1 router con buffer (coda) ∞:
  - non ci sono ritrasmissioni





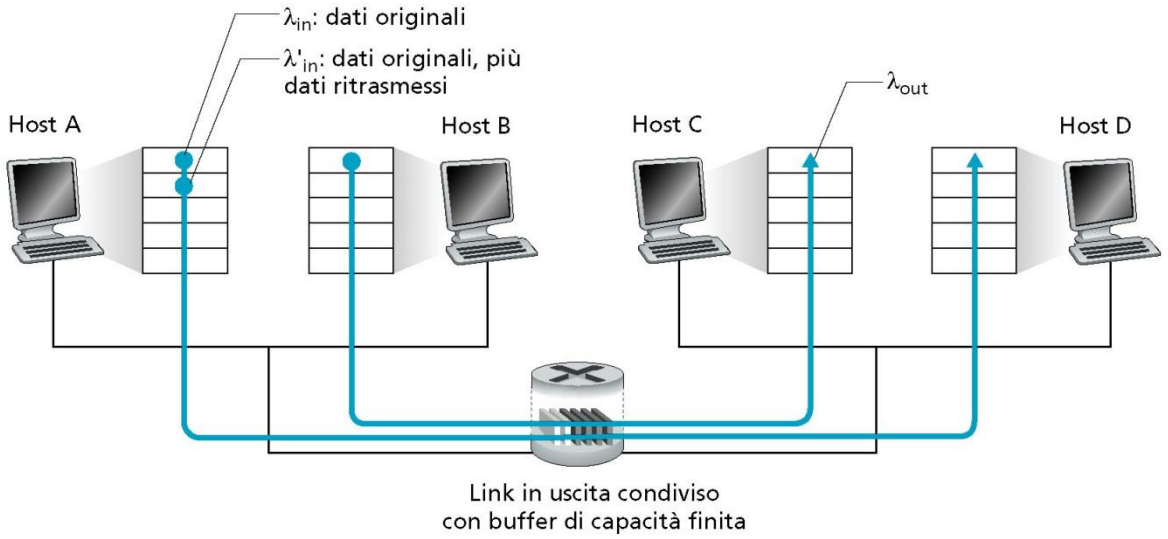


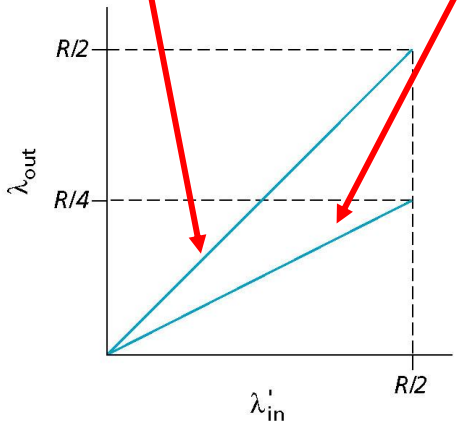
- I ritardi aumentano all'avvicinarsi del limite di capacità del canale
- Non si può superare il max throughput

Effetti della congestione: esempi (2/2)

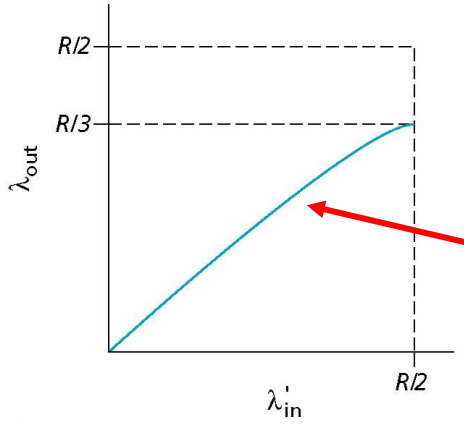
Il mittente invia dati solo quando il buffer non è pieno:
- caso ideale
  - ✓ no ritrasmissioni
  - ✓ throughput max = R/2

Scadenza prematura del timer del mittente:
- ✓ es: ogni segmento è spedito, in media, due volte
- ✓ throughput max = R/4





a.



b.

Il mittente rispedisce un segmento solo quando è sicuro che sia andato perso:

- ✓ il throughput effettivo è inferiore al carico offerto (trasmissioni dati originali + ritrasmissioni)
- ✓ es: curva in figura

Tecniche di Controllo della Congestione

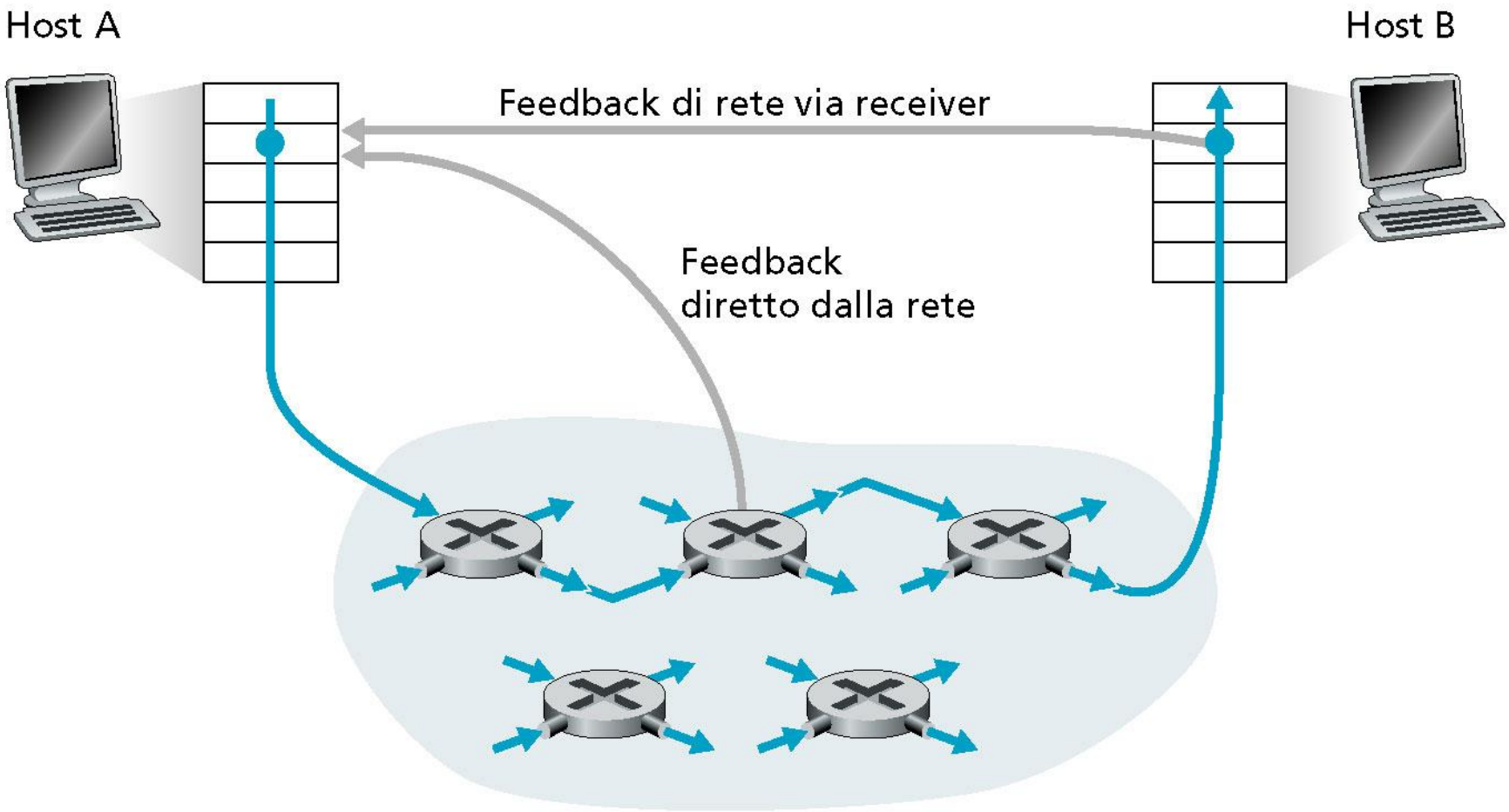
Approccio end-to-end:

- Nessuna segnalazione esplicita dalla rete
- A partire dall'osservazione di ritardi e perdite di pacchetti gli end-system deducono uno stato di congestione nella rete
- Approccio utilizzato da TCP

Approccio in base a segnalazione della rete:

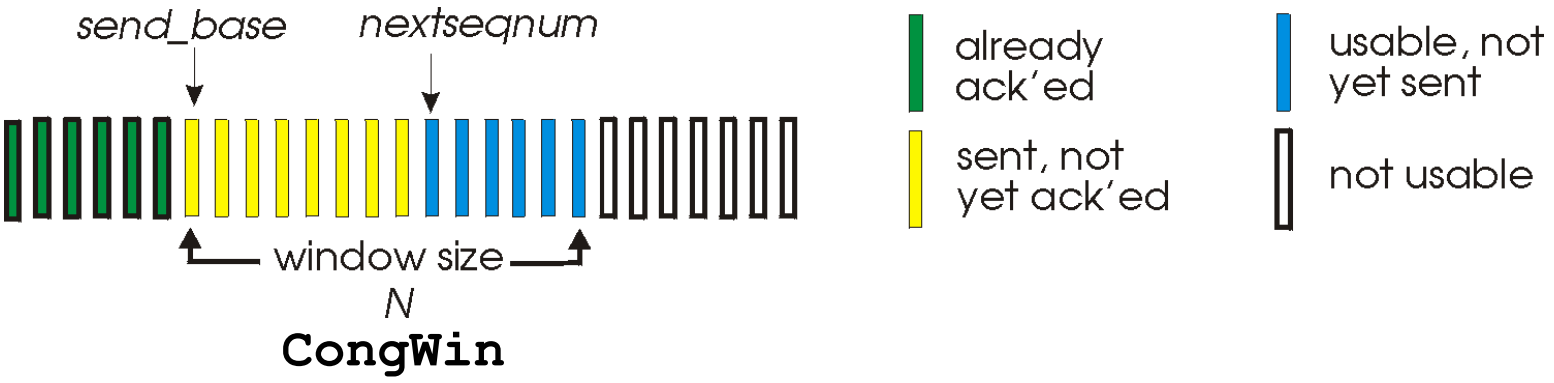
- I router forniscono informazioni circa lo stato della rete agli end-system:
  - l'invio di un singolo bit indica lo stato di congestione
    - SNA, DECbit, TCP/IP ECN, ATM
  - in alternativa, il sender è informato circa la massima frequenza alla quale può trasmettere

Feedback di rete: tecniche alternative



Controllo della congestione in TCP

- Controllo end-to-end: nessun feedback dalla rete
- Frequenza di trasmissione variabile:
  - dipendente dalla cosiddetta *finestra di congestione (CongWin)*



Considerando controllo di flusso e controllo di congestione insieme, si ha, dunque:

LastByteSent – LastByteAcked ≤ min{RcvWindow, CongWin}

Controllo della congestione: idea di base

- Si procede per tentativi, per stabilire quanto si può trasmettere:
  - obiettivo:
    - trasmettere alla massima velocità possibile (`Congwin` quanto più grande possibile) senza perdite
  - approccio utilizzato:
    - incrementare `Congwin` finché non si verifica la perdita di un segmento (interpretata come il sopraggiungere dello stato di congestione)
    - in seguito alla perdita di un segmento:
      - decrementare `Congwin`
      - ricominciare daccapo

Controllo della congestione: fasi

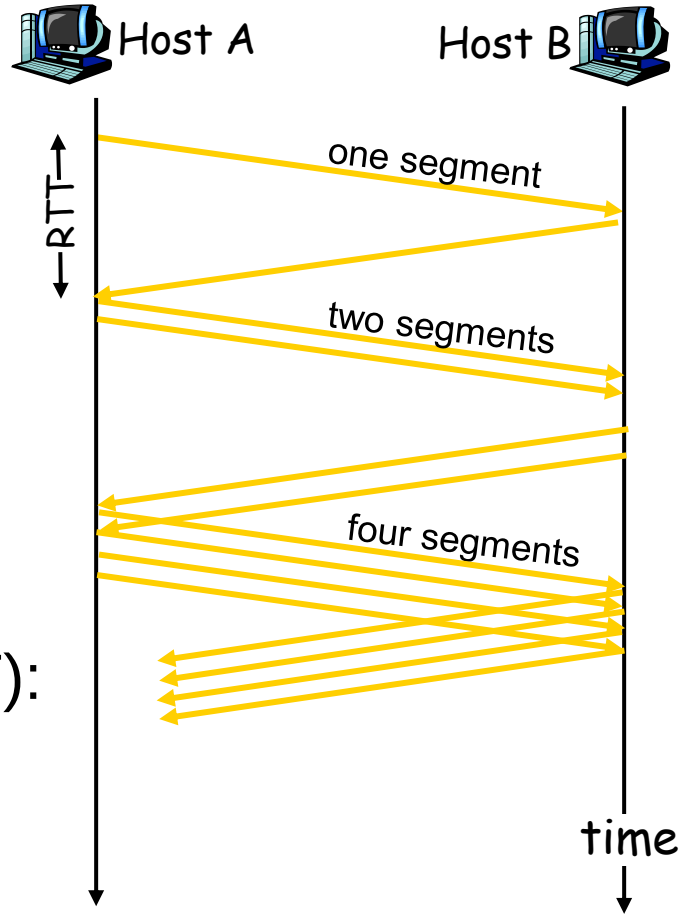
- *Slow Start*
  - Partenza lenta (per modo di dire!)
- Congestion Avoidance:
  - Additive Increase, Multiplicative Decrease (AIMD)
    - incremento additivo, decremento moltiplicativo

Lo Slow Start in TCP

Algoritmo Slowstart

```
//initialization
Congwin = 1 MSS

for (each segment ACKed)
      Congwin=Congwin+1MSS
until (loss event OR
        CongWin > threshold)
```



- Crescita esponenziale della dimensione della finestra (ogni RTT):
  - "Slow start" → termine improprio!
- Evento di *perdita*:
  - timeout
  - tre ACK duplicati consecutivi
  - Ritorna a CongWin = 1 MSS

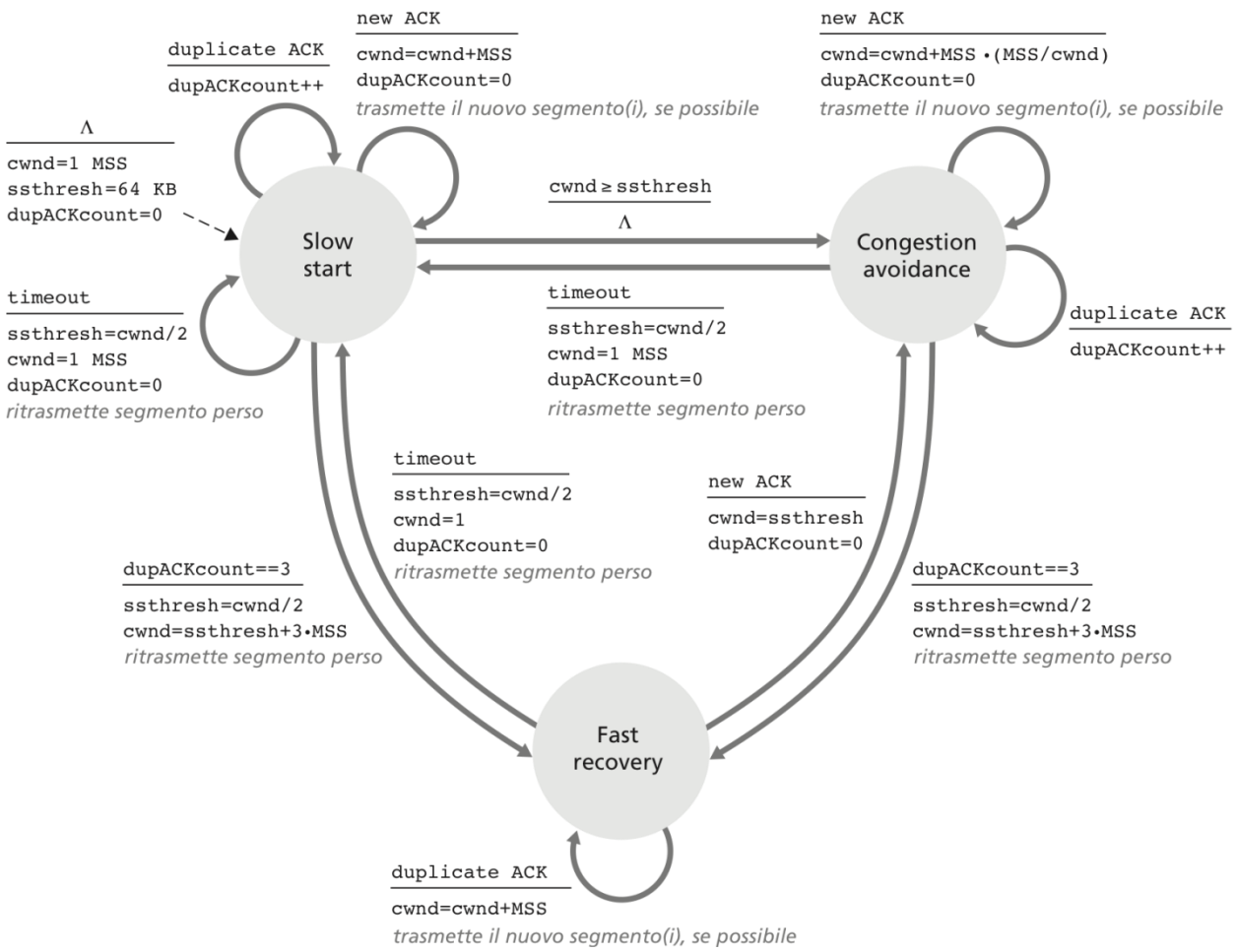
Dopo lo slow start: AIMD

- Una volta raggiunta la soglia, si entra in un comportamento detto ***Congestion Avoidance***:
  - ci si avvicina con cautela al valore della banda disponibile tra le due estremità della connessione (*Additive Increase, AI*)
    - incremento di CongWin di MSS·(MSS/Congwin) alla ricezione di un ACK
- Al sopraggiungere della congestione (*Multiplicative Decrease)*:
  - scadenza di un timeout → ***Slow Start***
    - Threshold = CongWin/2; CongWin = 1 MSS
  - ricezione di tre ACK duplicati consecutivi
    - Se TCP versione Tahoe, stessa cosa di timeout → ***Slow Start***
    - Se TCP versione Reno → ***Fast Recovery***
      - Threshold = CongWin/2; CongWin = Threshold+3MSS

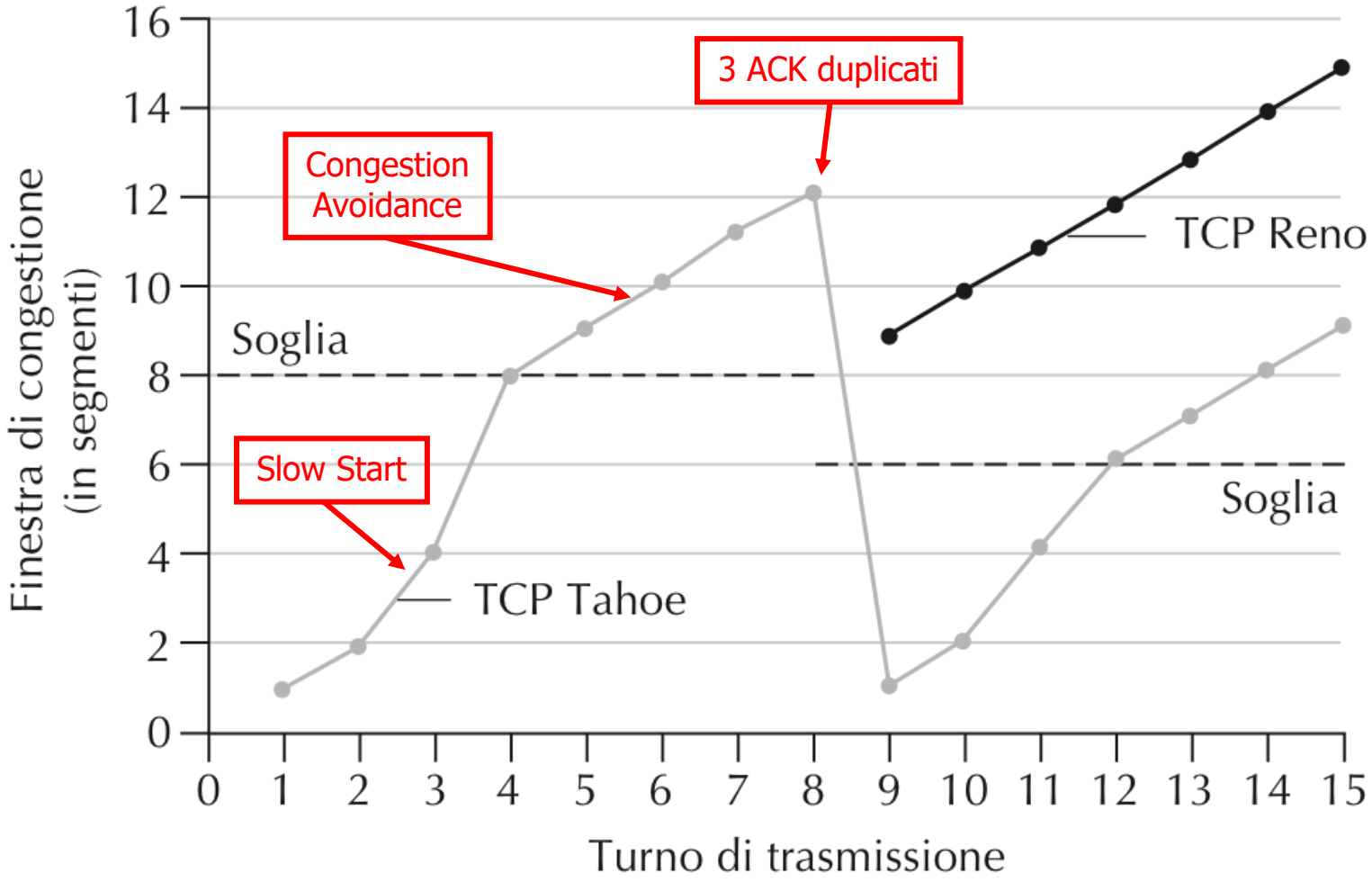
TCP Reno: "*fast recovery*"

- TCP Reno elimina la fase di Slow Start dopo un evento di perdita dedotto dalla ricezione di tre ACK duplicati:
  - tale evento indica che, nonostante si sia perso un pacchetto, almeno 3 segmenti successivi sono stati ricevuti dal destinatario:
    - a differenza del caso di timeout, la rete mostra di essere in grado di consegnare una certa quantità di dati
    - è possibile, quindi, evitare una nuova partenza lenta, ricominciando direttamente dalla fase di **fast recovery**

Controllo di congestione TCP: modello ASF



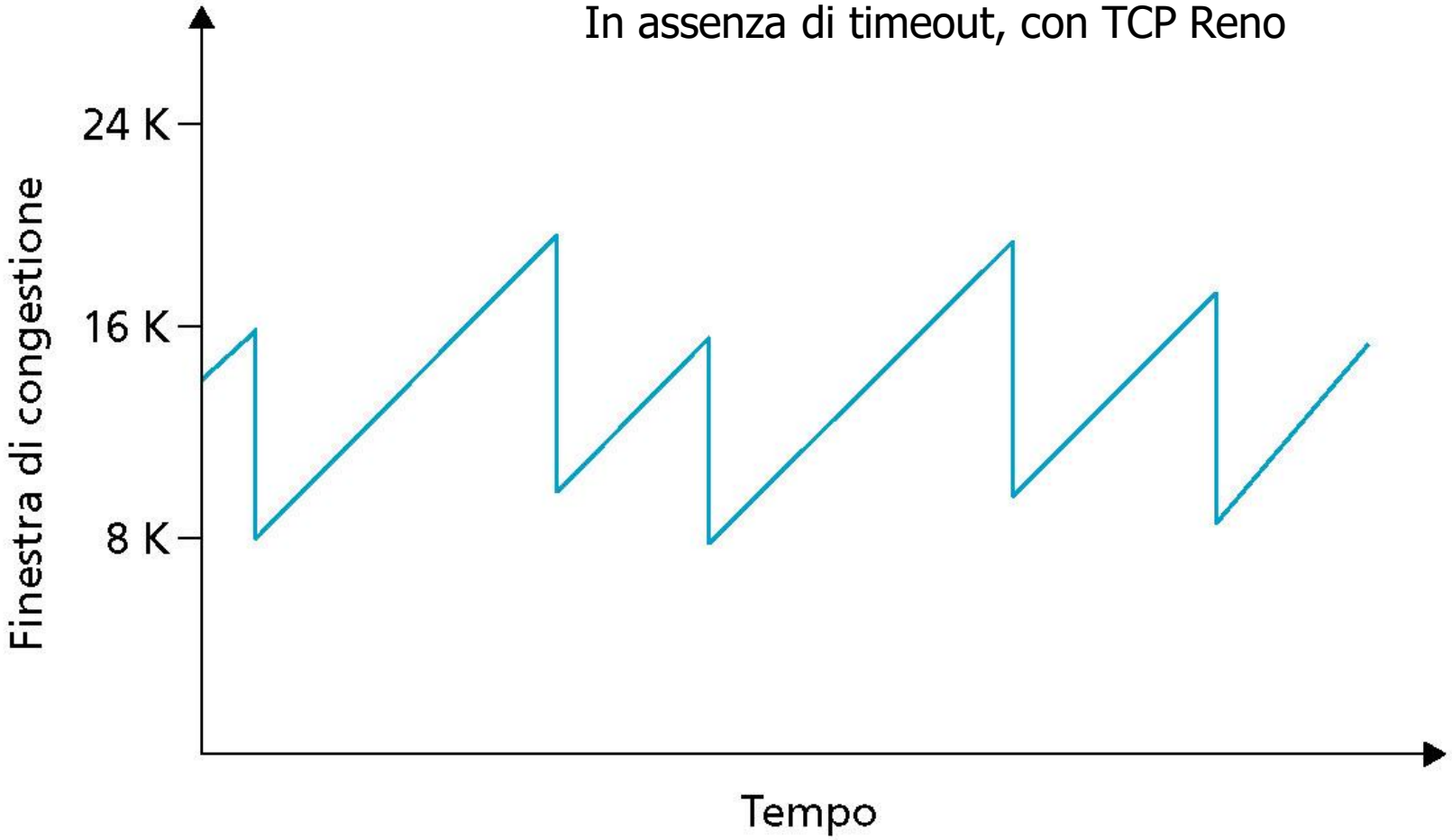
Controllo di congestione TCP: Tahoe vs Reno



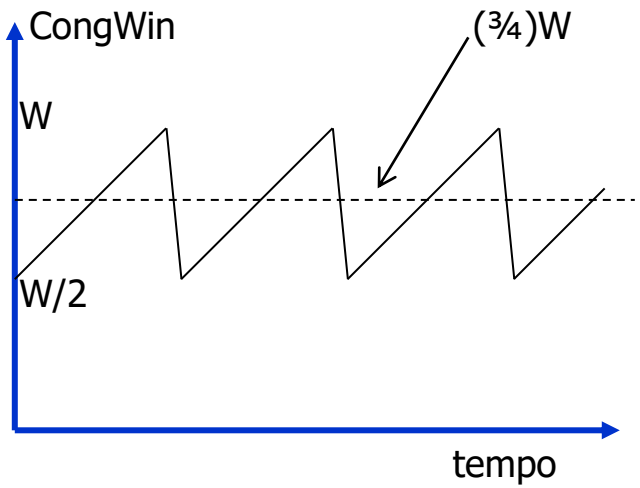
Ricapitolando…

- Finestra di congestione sotto la soglia:
  - Slow start
  - Crescita esponenziale della finestra
- Finestra di congestione sopra la soglia:
  - Prevenzione della congestione
  - Crescita lineare della finestra
- Evento di perdita dedotto da ACK duplicato 3 volte:
  - Soglia posta alla metà del valore attuale della finestra
  - TCP Reno:
    - Finestra posta pari alla soglia + 3 MSS
  - TCP Tahoe:
    - Finestra posta pari ad un segmento (MSS -- Maximum Segment Size)
- Evento di perdita dedotto da timeout:
  - Soglia posta alla metà del valore attuale della finestra
  - Finestra posta pari ad un segmento (MSS -- Maximum Segment Size)

AIMD: andamento a “dente di sega”



Throughput medio di una connessione TCP



Se il massimo valore di CongWin è costante (W) ed anche RTT è costante, il throughput medio di una connessione TCP è

$$T = \frac{0{,}75 \cdot W}{RTT} = \frac{W^*}{RTT}$$

dove W* = valore medio di CongWin

Se si desidera un throughput medio con deve essere:

$T = 10$ Gbps $= 10^{10}$ b/s $= 1250 \cdot 10^6$ byte/s

RTT = 100 ms

$W^* = T \cdot RTT = 1250 \cdot 10^6 \cdot 10^{-1}$ byte $= 125 \cdot 10^6$ byte

Se 1 MSS = 1500 byte, esprimendo la dimensione media in MSS, si ha:

$W^* = (1250/1500) \cdot 10^5$ MSS $= 83333$ MSS

Throughput medio di una connessione TCP (2)

- La stima del throughput di una connessione TCP è stata oggetto di diversi studi
- La formula semplificata

$$T = \frac{0.75 \cdot W}{RTT}$$

ignora l'effetto della perdita di pacchetti

- In (*) è presentato un modello leggermente più accurato che tiene in conto della perdita di pacchetti
- Detto L il tasso medio di perdita di un pacchetto, è possibile ricavare:

$$T = \frac{C \cdot W}{RTT \cdot \sqrt{L}}$$

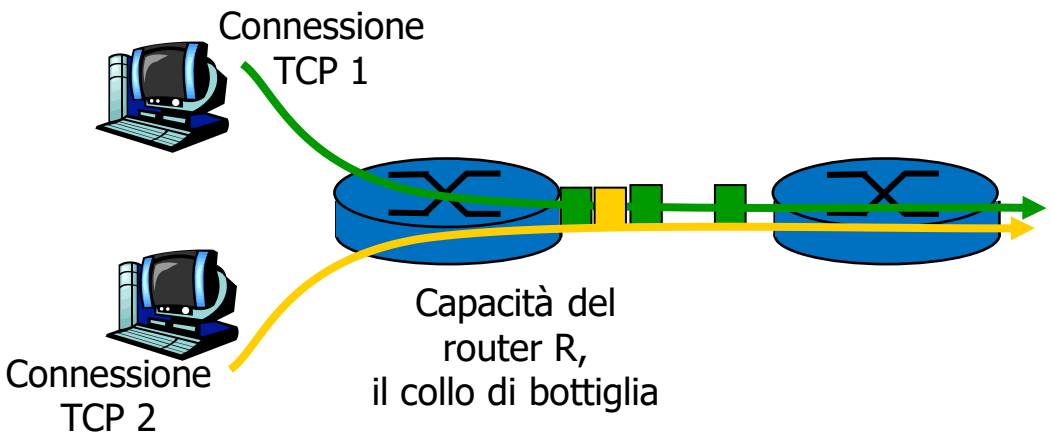
con $C = 1.22 = \sqrt{3/2}$

(*) Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis.
The macroscopic behavior of the TCP congestion avoidance algorithm.
ACM SIGCOMM Computer Communications Review, vol. 27, issue 3, July 1997, pp. 67-82

Equità tra le connessioni TCP (1)

Se K sessioni TCP condividono lo stesso collegamento con ampiezza di banda R, che è un collo di bottiglia per il sistema, ogni flusso dovrebbe avere un tasso di trasmissione medio pari a R/K

equità (fairness)



Si può dire che AIMD è equo?

Equità tra le connessioni TCP (2)

- Ipotesi:
  - MSS e RTT uguali per le due connessioni:
    - a parità di dimensioni della finestra quindi il throughput è lo stesso
  - Entrambe le connessioni si trovano oltre lo slow start:
    - fase di prevenzione della congestione:
      - incremento additivo
      - decremento moltiplicativo