- Politecnico di Milano
- Dipartimento di Elettronica e Informazione

- 7 -
Tecniche di filtraggio del traffico

Laboratorio di Reti di Telecomunicazione

Caratterizzazione del traffico IP

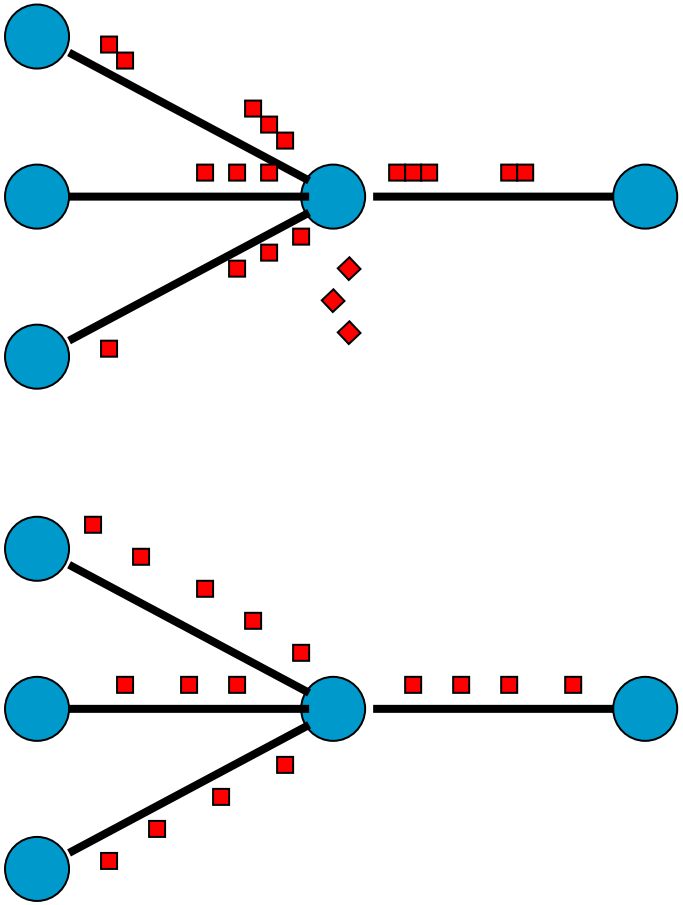
- Per caratterizzare un flusso di traffico si possono usare diversi parametri: la frequenza media e di picco, la distribuzione statistica dei tempi di interarrivo o della lunghezza dei pacchetti, ecc.

- Il traffico generato dalle sorgenti nelle reti IP presenta un alto grado di variabilità e imprevedibilità.
  - Il traffico in uscita non ha un andamento costante, bensì a **burst;** la sorgente alterna periodi di inattività a sequenze di pacchetti ravvicinati.
  - Nel caso di traffico bursty, la banda è normalmente definita in termini di frequenza di picco, frequenza media e durata dei burst.

Traffico bursty e congestione

- **Il traffico bursty rende complessa la gestione e la pianificazione della rete provocando più facilmente congestione nella rete.**

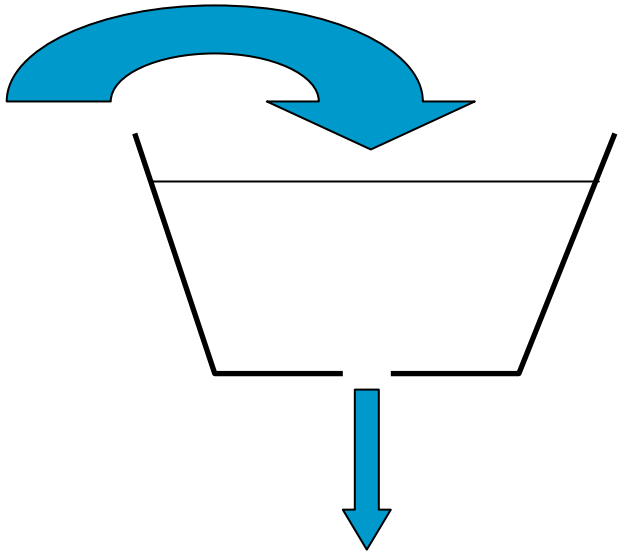
- **Traffic Shaping**
  - **Tecnica per modellare il flusso di traffico all'interno di forme o schemi più semplici da descrivere e controllare**
  - **Es: Leaky bucket e Token bucket.**

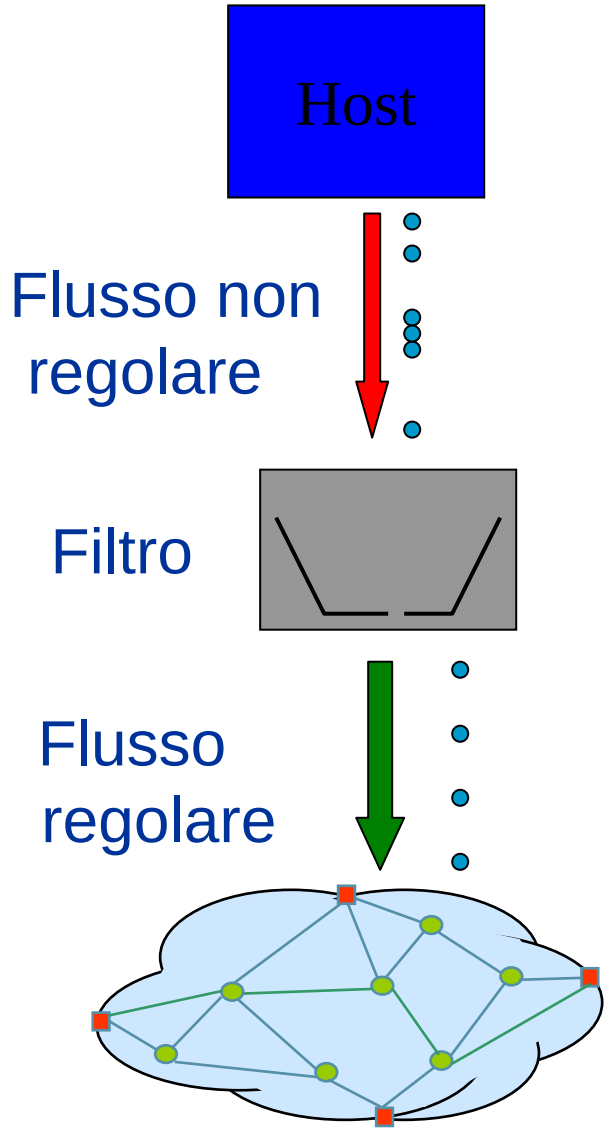


Traffic shaping

- Allevia la burstiness del traffico forzando i pacchetti ad essere trasmessi secondo un ritmo più regolare.
- Per esempio è usato nelle reti ATM.
- Regola la frequenza media di trasmissione.
- E' un meccanismo di controllo di congestione ad anello aperto
  - Al contrario il meccanismo del TCP è ad anello chiuso e regola la quantità di dati in transito.
- Semplifica il service agreement tra utente e gestore di rete.
  - Importante nelle applicazioni real-time come audio, video.

Leaky Bucket 1

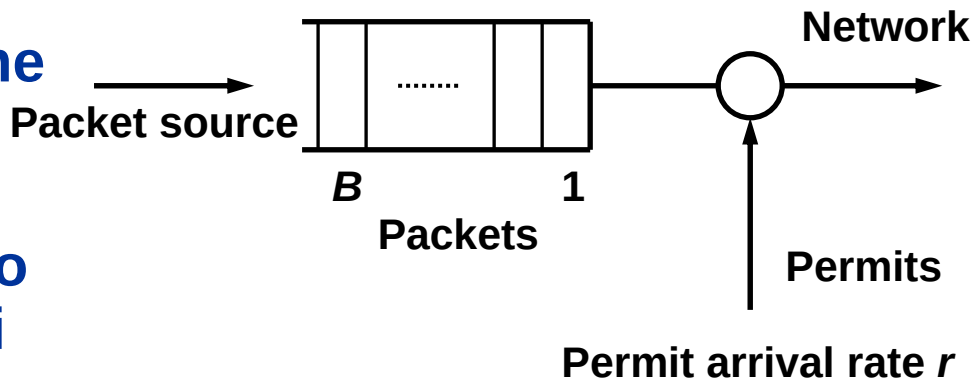




1. Non conta la velocità di ingresso dell'acqua, la portata in uscita non può superare il valore limite.
2. Quando il secchio è pieno, l'acqua trabocca e viene persa

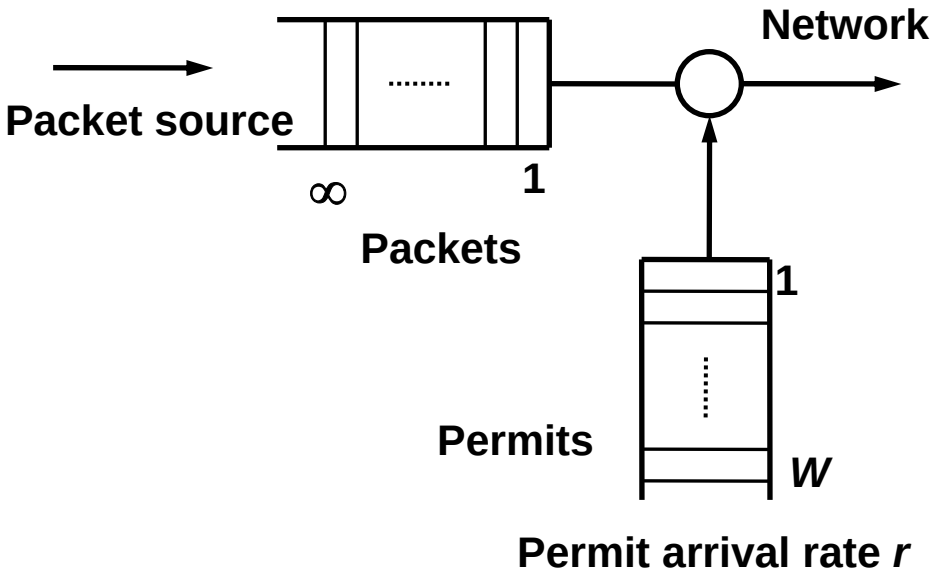
Leaky Bucket 2

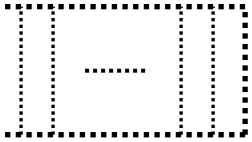
- E' equivalente ad un sistema a coda a servente singolo con tempo di servizio costante.
- Parametri: lunghezza della coda *B* e frequenza di uscita *r*
- Le dimensioni della coda:
  - determinano quanto bursty può essere il traffico in entrata,
  - determinano insieme con la frequenza *r* il massimo ritardo di accodamento, *B*/*r*.
- Pacchetti a dimensione fissa (es. celle ATM): pacchetto utilizzato come unità.
- Pacchetti a dimensione variabile (es. IP): numero fisso di byte per unità di tempo.



Token Bucket 1

- I pacchetti in coda nel data buffer necessitano di un token per essere trasmessi. L'accumulo dei token nei periodi inattivi permette di smaltire meglio eventuali burst di traffico.
- Parametri: lunghezza della coda dei permessi *W* e frequenza di arrivo dei permessi *r*.
- La frequenza di uscita dei pacchetti dipende dal numero di token disponibili.





- Il buffer dei dati in ingresso ha lunghezza infinita, ma può essere seguito da un buffer di dimensione finita prima dell'ingresso in rete.

Token Bucket 2

- I token si accumulano in una coda di lunghezza *W* con frequenza di ingresso costante pari a *r*.
- I pacchetti da trasmettere si accodano nel buffer dei dati e vengono trasmessi solo se ci sono token nella coda dei permessi.
- Se non ci sono token nella coda dei permessi, non vengono trasmessi pacchetti.
- Quando la coda dei permessi è piena, ulteriori token vengono scartati.
- La burstiness del traffico è limitata superiormente perché non potranno mai essere inviati più di $W + r\,\Delta t$ pacchetti in un intervallo di tempo Δt e la frequenza di trasmissione nel lungo periodo non eccede *r*.

Leaky versus Token Bucket

- Leaky bucket: la sorgente filtrata o è inattiva o trasmette pacchetti a frequenza costante.

- Token bucket:
  - si accumulano crediti quando la linea è inutilizzata (cioè quando non ci sono pacchetti in coda);
  - quando arriva un burst di traffico il “credito” accumulato permette di inviare i pacchetti in uscita ad una frequenza maggiore, quindi il buffer della coda in entrata viene liberato prima
  - se misurata sul lungo periodo, la frequenza media è comunque mantenuta su un valore predefinito (pari alla frequenza di rilascio dei token)

Token bucket in *ns*

- In *ns* è implementata una versione di "token bucket filter" avente dimensione *B* del buffer di pacchetti (ovviamente) finita; ci si riconduce al caso classico prendendo *B* sufficientemente grande

- Il filtro viene creato in questo modo:

```
set tbf0 [new TBF]
```

- Questo oggetto ha tre parametri fondamentali: la frequenza di arrivo dei permessi e la dimensione delle code dei permessi e di pacchetti; ad esempio:

```
$tbf0 set bucket_ 1Mb
$tbf0 set rate_ 200kb
$tbf0 set qlen_ 50
```

coda dei permessi di 1 Mbit
frequenza di arrivo dei permessi: 200 kbps
dim. della coda di pacchetti (50 pacchetti)

- (attenzione alle unità di misura!)

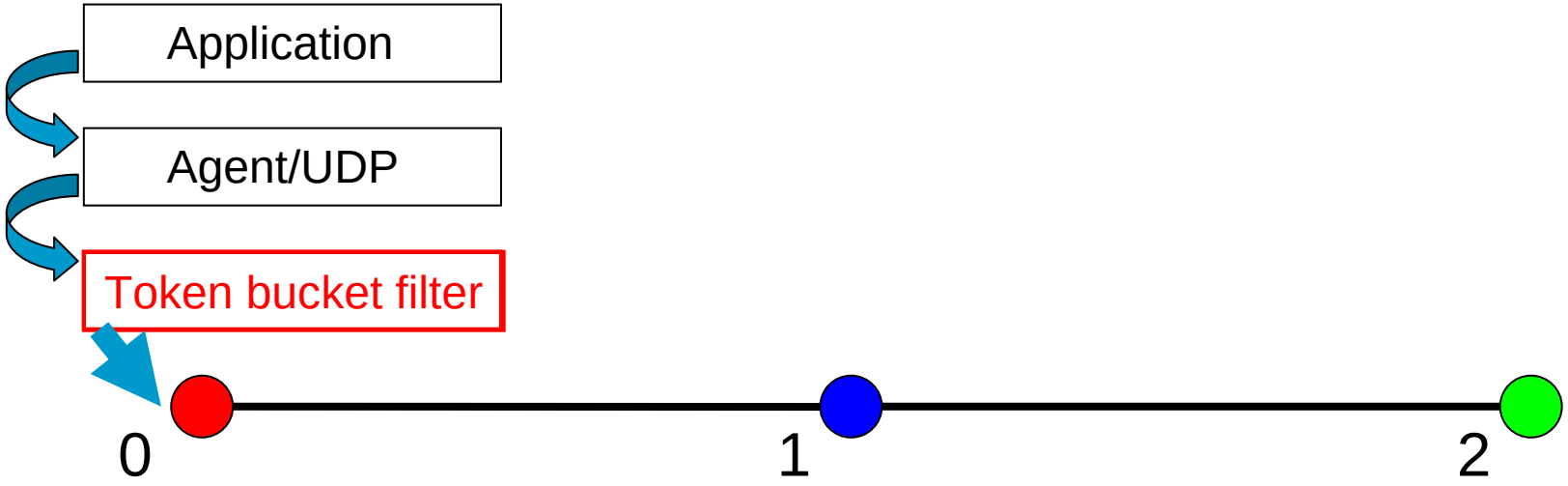
Leaky bucket in *ns*

- Il leaky bucket non è altro che un token bucket in cui
  - la dimensione di coda dei permessi *è pari ad un pacchetto*
  - la dimensione della coda dei pacchetti è *finita*
- Esempio di definizione di un leaky bucket:

```
set pSize 1000                  ;# pacchetti di 1000 byte
set udp0 [new Agent/UDP]
$udp0 set packetSize_ $pSize
set leaky0 [new TBF]
$leaky0 set bucket_ [expr 8*$pSize] ;# da byte a bit
$leaky0 set rate_ 200kb
$leaky0 set qlen_ 50
```

Inserimento di filtri 1

- **Come fare a posizionare un filtro nel punto desiderato lungo un flusso di traffico?**

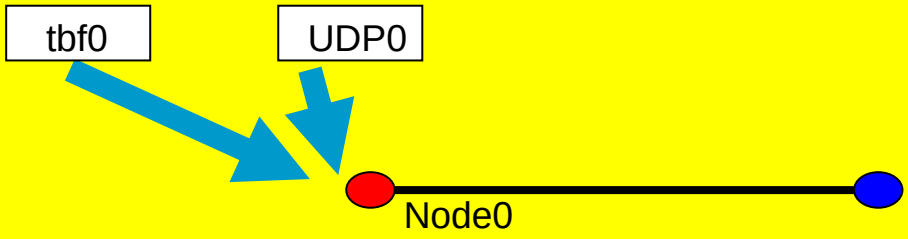


Inserimento di filtri 2

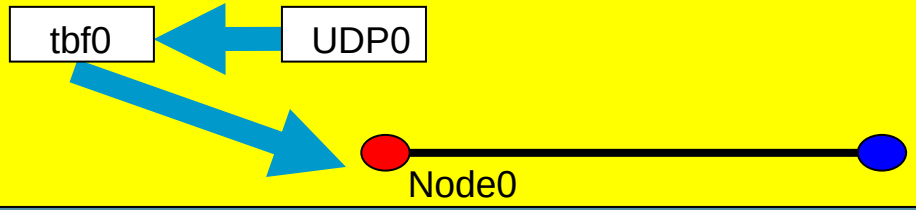
- E' possibile farlo utilizzando il metodo `target` comune a numerosi oggetti di *ns:*

```
$ns attach-agent $Node0 $UDP0 ;# collega l'uscita di UDP0 a Node0
```


```
set tg [$UDP0 target] ;# assegna alla variabile tg l'ID dell'og-
                      ;# getto in uscita da UDP0 (il Node0)
$tbf0 target $tg      ;# invia sull'oggetto tg (Node0) il
                       ;# traffico in uscita da tbf0
```




```
$UDP0 target $tbf0 ;# invia su tbf0 il traffico in uscita da UDP0
```

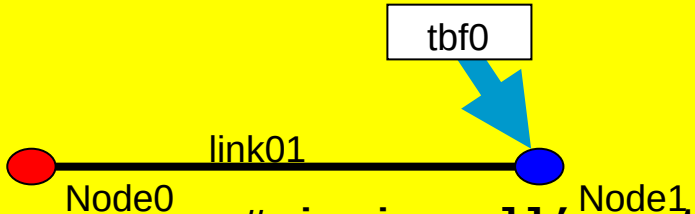


Inserimento di filtri 3

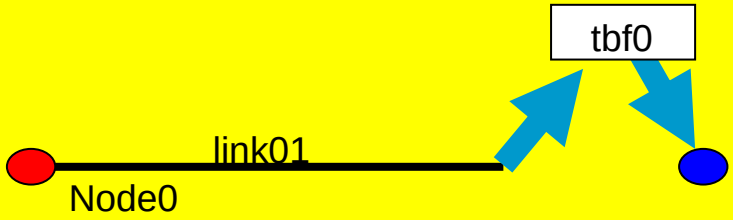
- Mediante il metodo `target` si può anche mettere una cascata di più filtri o inserire un filtro all'uscita di un link:

```
$ns duplex-link $Node0 $Node1 1Mb 100ms DropTail
[…]
set link01 [[$ns link $Node0 $Node1] set link_]
                                ;# assegna a link01 l'ID del
                                ;#  link tra Node0 e Node1

set e01 [$link01 target]        ;# assegna a e01 l'uscita
                                ;#  del link 0-1
```




```
$tbf0 target $e01               ;# invia sull'oggetto e01 (Node1)
                                ;#  il traffico in uscita da tbf0
```

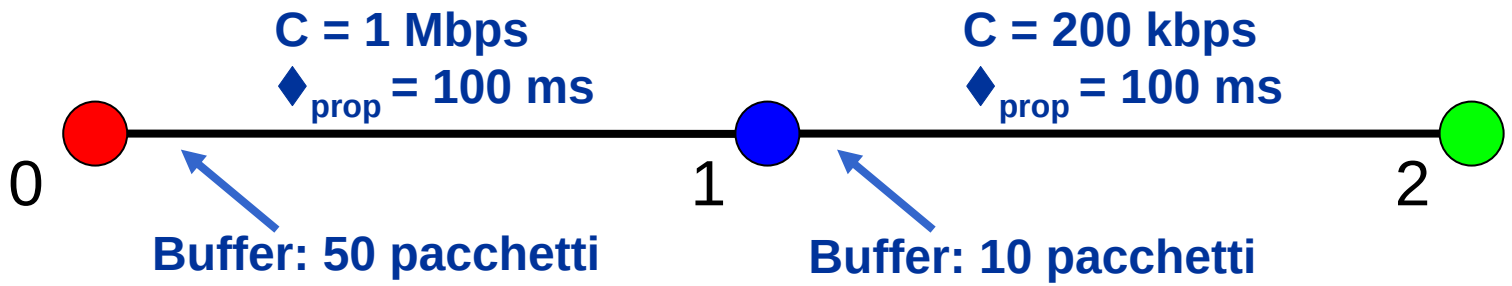



```
$link01 target $tbf0 ;# invia su tbf0 il traff.in uscita dal link
```

Esercizio 12

a. Si consideri la rete in figura:

Durata sim.: 5 s



- Al nodo 0 è presente una sorgente esponenziale ON/OFF che genera pacchetti di 1000 byte con T_{ON} medio di 0.01 s, T_{OFF} medio di 0.15 s e frequenza (durante il periodo ON) di 2 Mbps; la sorgente è attiva fino all'istante t=3 s
- Con un agente LossMonitor in ricezione al nodo 2, misurare la percentuale di pacchetti persi al buffer del nodo 1

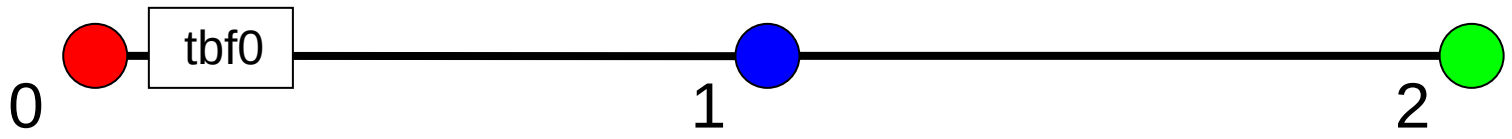
b. Utilizzando un filtro opportuno al nodo 0, limitare la frequenza massima sul link 0-1 a 500 kbps

- Confrontare la percentuale di pacchetti persi al nodo 1 con il caso precedente

Esercizio 13a (shaper)

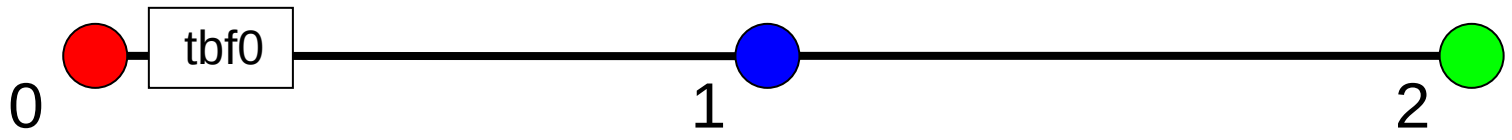
- Si consideri la rete della figura precedente, senza filtro di traffico
- Utilizzando un (diverso) filtro al nodo 0, ottenere sul link 0-1 il profilo di traffico seguente:
  - MBS (Maximum Burst Size) di 5 pacchetti
  - Frequenza di picco non superiore alla capacità del link 0-1 (1 Mbps)
  - Frequenza media pari alla capacità del link 1-2 (200 kbps)

- Qual è la minima dimensione del buffer al nodo 1 per non avere perdite *in quel nodo*?



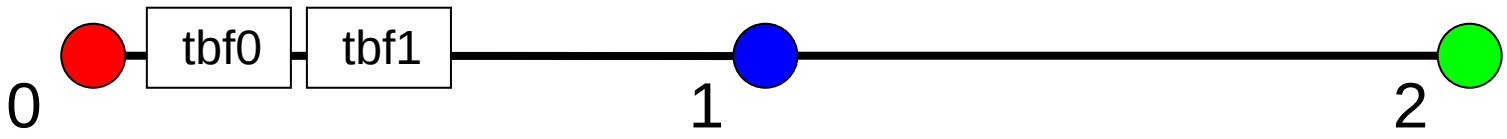
Esercizio 13a (shaper)

- **Si confronti il caso di traffico filtrato (es. 13a) e non filtrato (es. 12a) con *nam*; si esamini l'azione del filtro nei grafici temporali della banda occupata sul link 0-1**
  - **dopo aver eseguito la simulazione, si richiami *nam***
  - **eventualmente si modifichi il layout dei nodi in modalità edit**
  - **si torni in modalità view e si faccia partire l'animazione (play)**
  - **cliccando col tasto destro del mouse sul link 0-1, si selezioni *graph bandwidth 0→1***
  - **In una striscia sotto lo schema della rete viene visualizzato il grafico temporale richiesto; la massima ordinata corrisponde alla capacità del link**



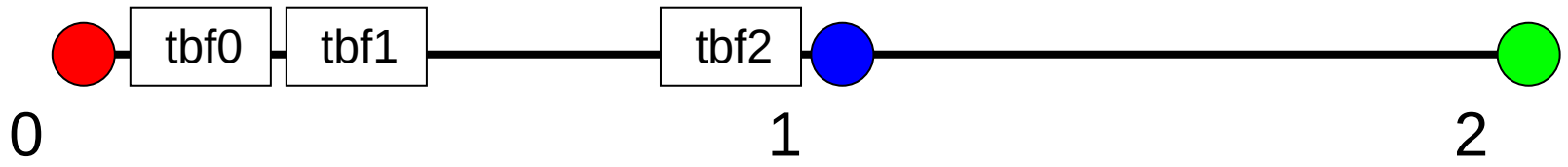
Esercizio 13b (shaper)

- Aggiungendo un secondo filtro prima del link 0-1, limitare la frequenza di picco in uscita dal nodo 0 a 210 kbps
- Qual è adesso la minima dimensione del buffer al nodo 1 per non avere perdite in quel nodo?
- Visualizzare in *nam* l'azione del filtro



Esercizio 13c (policer)

- **Aggiungendo un terzo filtro prima del nodo 1, limitare la frequenza media a 200 kbps**
- **Utilizzando un buffer di *2* pacchetti al nodo 1 (buffer minimo), vengono persi dei pacchetti?**



Riferimento: http://www.run.montefiore.ulg.ac.be/HTML/Cours/Multimedia/tp2-tbf/tp2-tbf.html