Control de congestión y buffers

Area de Ingeniería Telemática
http://www.tlm.unavarra.es

Grado en Ingeniería en Tecnologías de Telecomunicación, 4º

Rate- vs window- control

Rate- vs Window- control

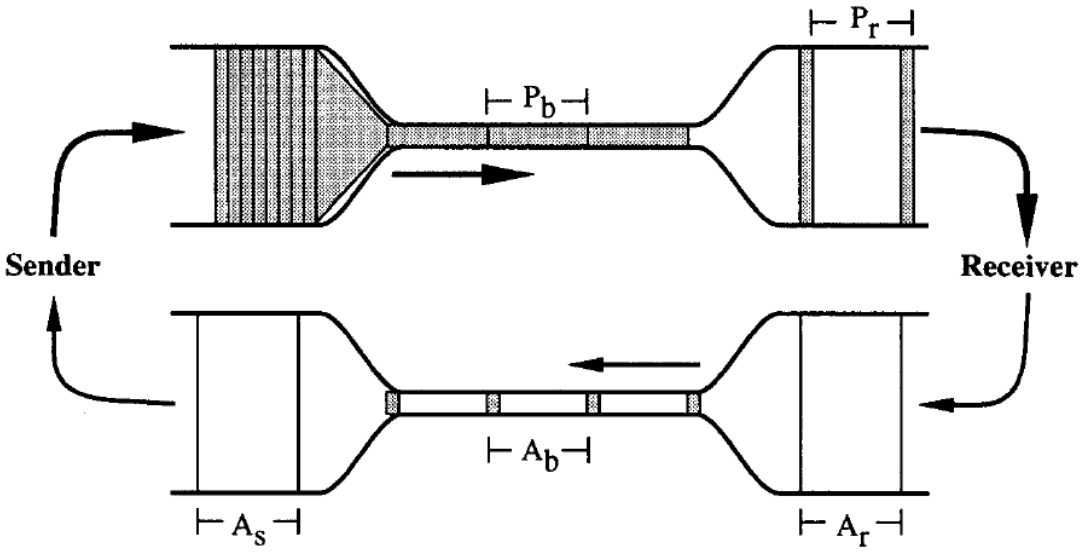
- Dos métodos de control del envío datos:
  - Control basado en la tasa de envío
  - Control basado en ventana

Rate- control

- Control basado en la tasa de envío
- De alguna forma el emisor sabe a qué tasa debe enviar
- Le informa el receptor o un equipo intermedio
- Simple
- Más apropiado para streaming
  - No queremos que se detenga pues la fuente no lo hace
  - Mantendría la tasa ante congestión
  - Si se le notifica de la nueva tasa adecuada podría hacer transcodificación

Window- control

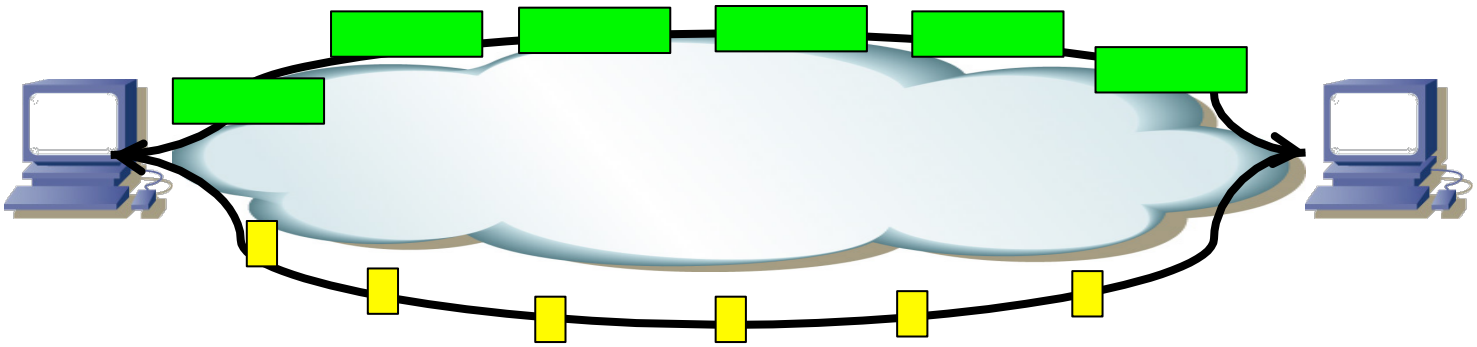
- Control basado en ventana
- Hay un máximo de paquetes que pueden estar “en vuelo”
- Solo se puede introducir uno cuando otro ha salido/llegado
- Receptor notifica a emisor de que un paquete ha llegado
- Más conservador pues deja de enviar si paquetes “no salen”
- Self-Clocking: la tasa de envío está limitada por la de ACKs



V. Jacobson, Proc. ACM SIGCOMM’88

Window- control

- La ventana debe ser lo suficientemente grande para sacar provecho a todo el canal
- Eso es un tamaño al menos BW x RTT
- Eso requiere una buena estimación del RTT



Window- control

- La transferencia se autorregula
- ¿Entonces no puede haber ráfagas?
- (...)

Ráfagas

- Ejemplo:
  - Emisor envía y los paquetes se encolan en un puerto de salida
  - Cuando se transmiten salen juntos a la tasa de ese enlace
  - Puede ser de alta capacidad y solo haber sufrido congestión temporal
  - Los ACKs de paquetes que llegan próximos saldrán próximos
  - Y provocarán que se “abra” la ventana de golpe y el emisor pueda enviar varios paquetes a la tasa de su interfaz

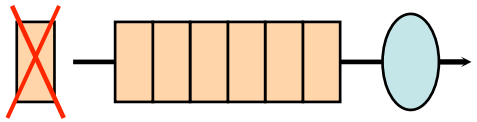
Ráfagas

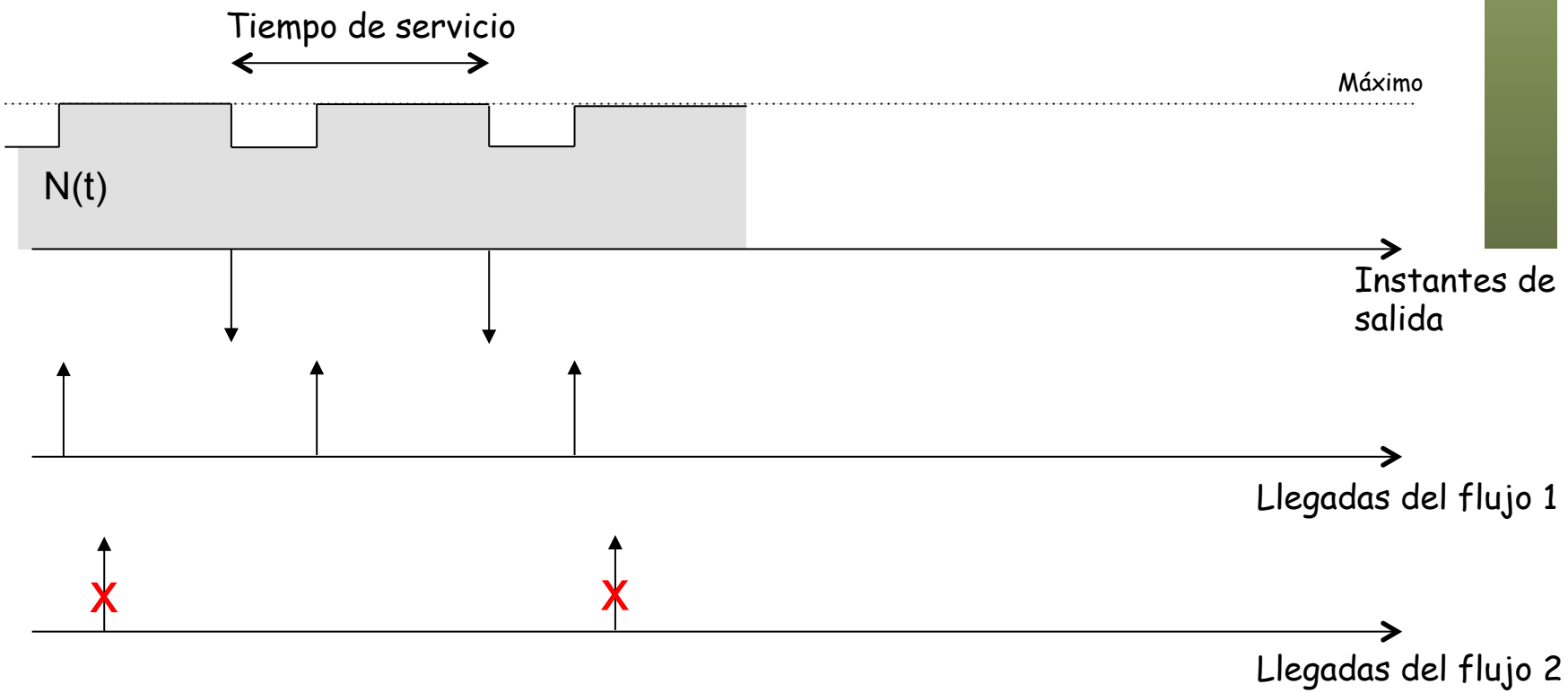
- **Otro ejemplo:**
  - Ha habido una transferencia, que ha podido estar bien auto-regulada
  - Pero ya no hay más datos a transmitir y se han confirmado todos
  - La ventana queda completamente “abierta”
  - Ahora si la aplicación genera una gran cantidad de datos se podrán enviar tantos como la ventana en una ráfaga

Buffers y sincronización

Traffic phase effects

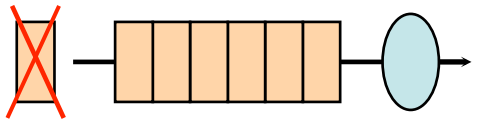
- Un ejemplo
- Todos los paquetes de igual tamaño
- Enlace congestionado, cola llena
- Flujo 1 ocupa el hueco que queda al terminar de transmitir
- Otra vez (llegadas periódicas)
- Llegada del flujo 2 se pierde (...)

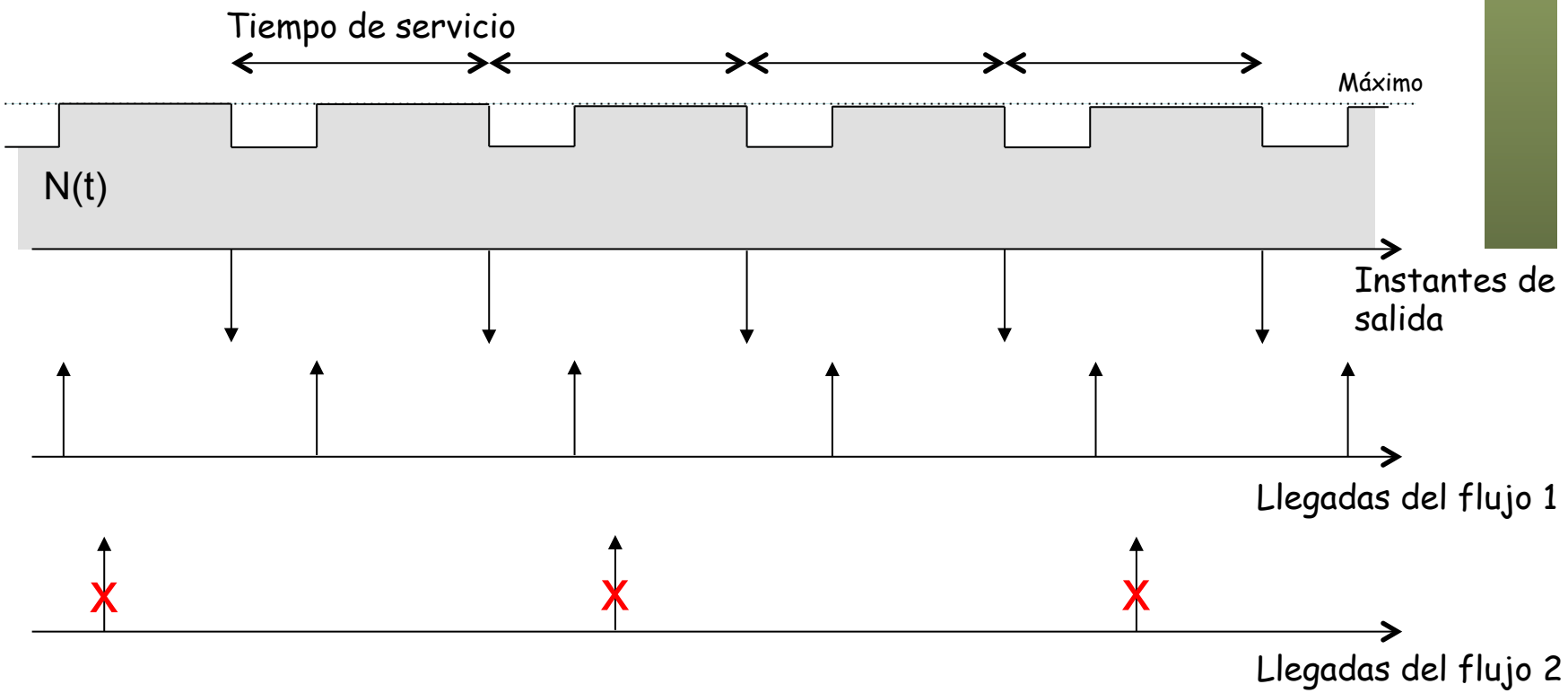




Traffic phase effects

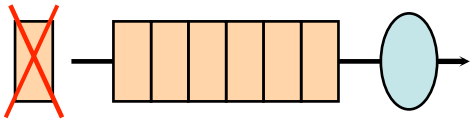
- Flujo 1 ocupa el hueco que queda al terminar de transmitir (otra vez)
- Otra vez
- Otra vez llegada del flujo 2 se pierde (periódicas)
- ¡ Se van a perder siempre !
- Salidas periódicas (siempre hay *backlog* y con paquetes de tamaño constante)



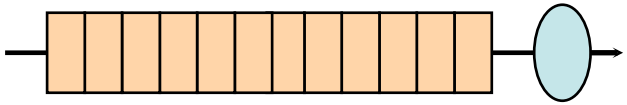


Romper la sincronización

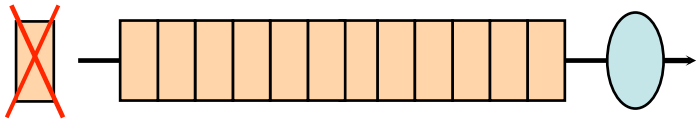
- Una de las causas son las colas drop-tail
- Y que detectamos congestión con pérdidas



- Evitaríamos pérdidas con cola más grande

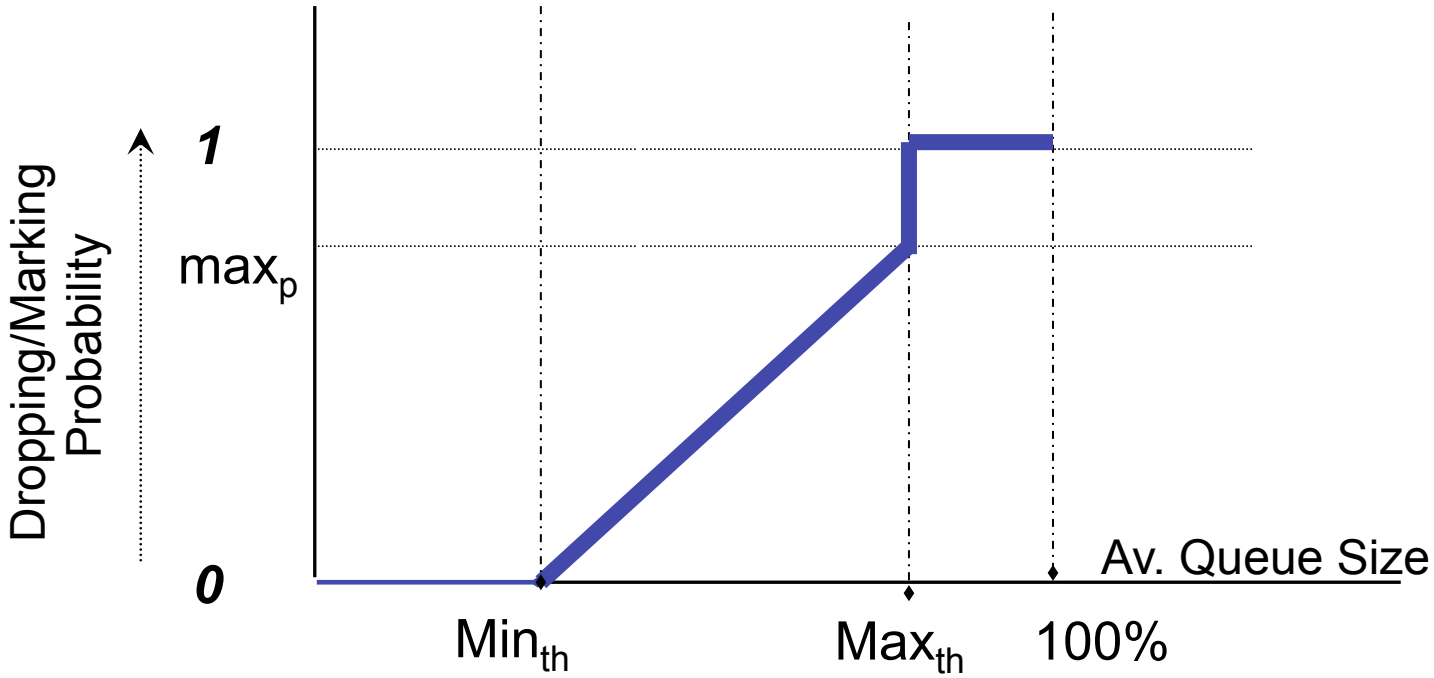


- ¿Resuelve el problema?
- No, no evita la congestión
- Sin pérdidas no hay *feedback* de que hay congestión
- Luego las fuentes siguen aumentando la tasa de envío
- Y acaba habiendo pérdidas



Romper la sincronización

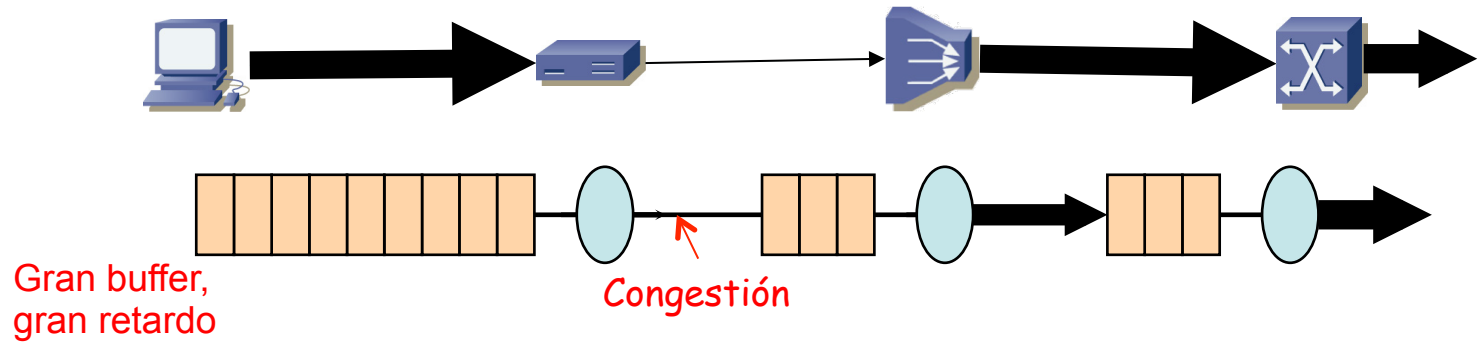
- Introducir aleatoriedad en las pérdidas
- AQM, típicamente RED (Random Early Detection)
- Permite romper la sincronización y mantener la cola poco ocupada
- Complejo de configurar



Tamaño de los buffers

Tamaño de buffers

- ¿Bueno, pero en realidad queremos el buffer ocupado para conseguir mayor throughput, no?
- Aún con un buffer grande habrá pérdidas
- Peor aún: además hay mayor retardo
- De hecho hoy en día se están poniendo buffers muy grandes
- Eso en enlaces congestionados (por ejemplo upstream ADSL) lleva a grandes retardos (centenares de ms o incluso segundos)



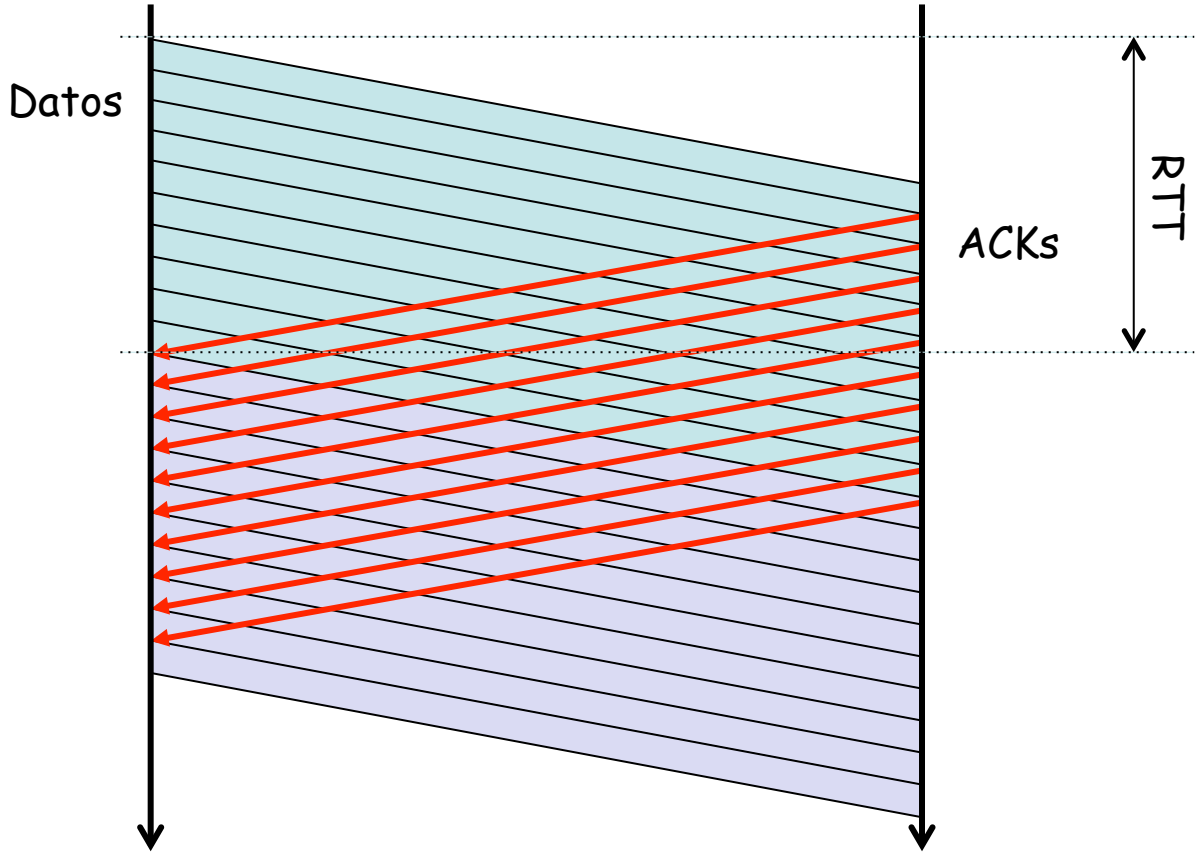
- Aumenta el RTT y con él la pérdida de interactividad (resolución de DNS, comienzo de conexiones cortas, etc)

¿Tamaño del buffer?

- Queremos buffer para absorber la intermitencia del tráfico
- No lo queremos grande para que no incremente el retardo
- Retardo grande aumenta el tiempo que tarda el cliente en recibir notificaciones de congestión
  - Por ejemplo el RTO calculado será mayor
  - Entonces tarda más tiempo en reconocer una pérdida y reaccionar
- ¿Cuál es el tamaño ideal de buffer?

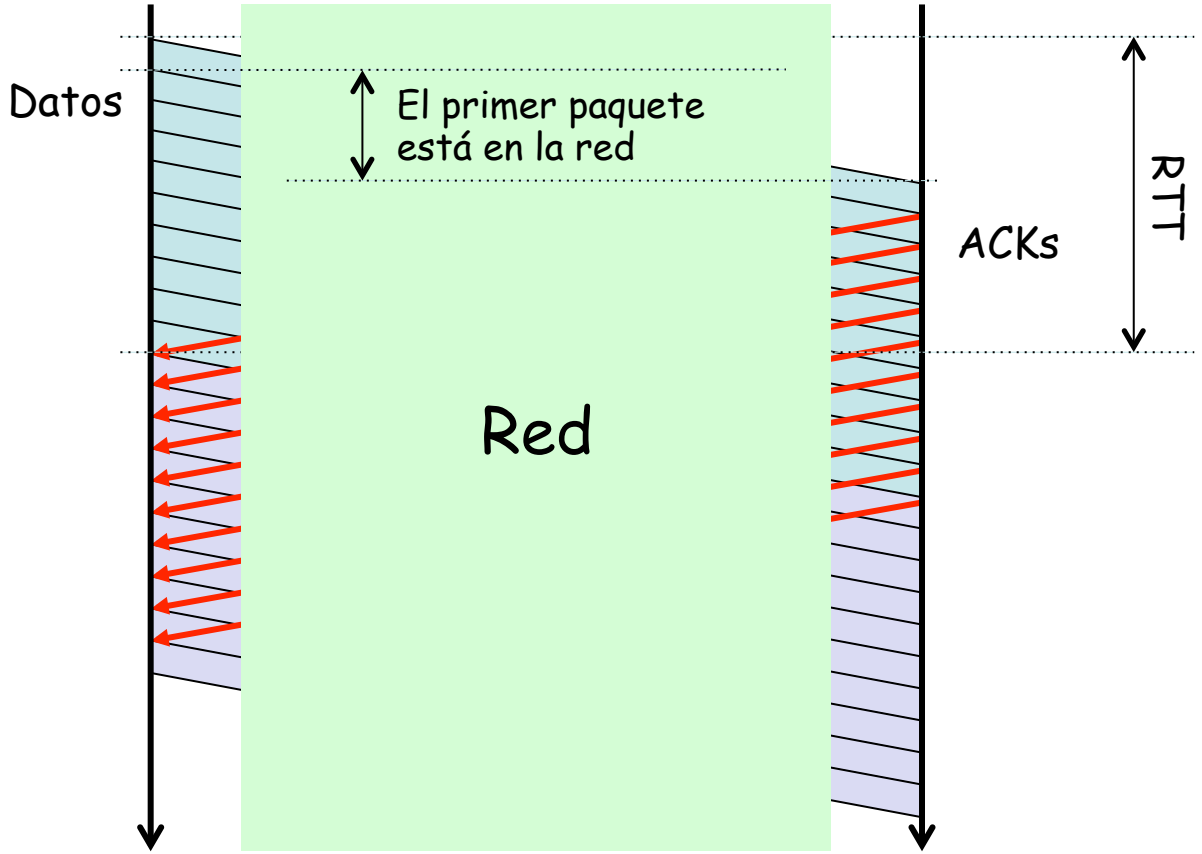
Buffer

- Transporte window-based
- Ventana para saturar = RTT x Capacidad
- (…)



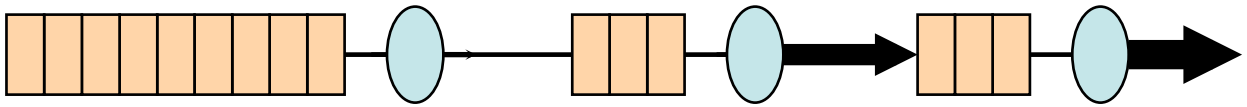
Buffer

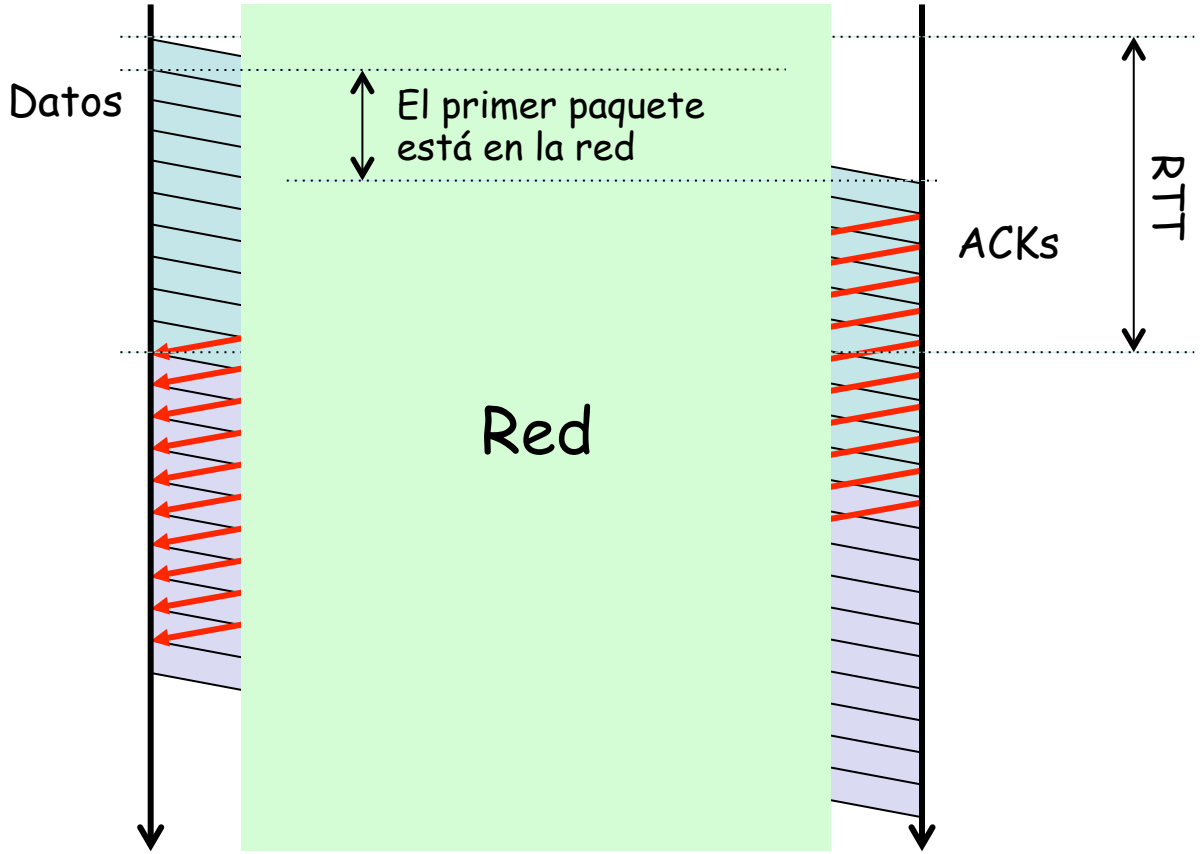
- Transporte window-based
- Ventana para saturar = RTT x Capacidad
- ¿Dónde están los paquetes?
- En la red (…)



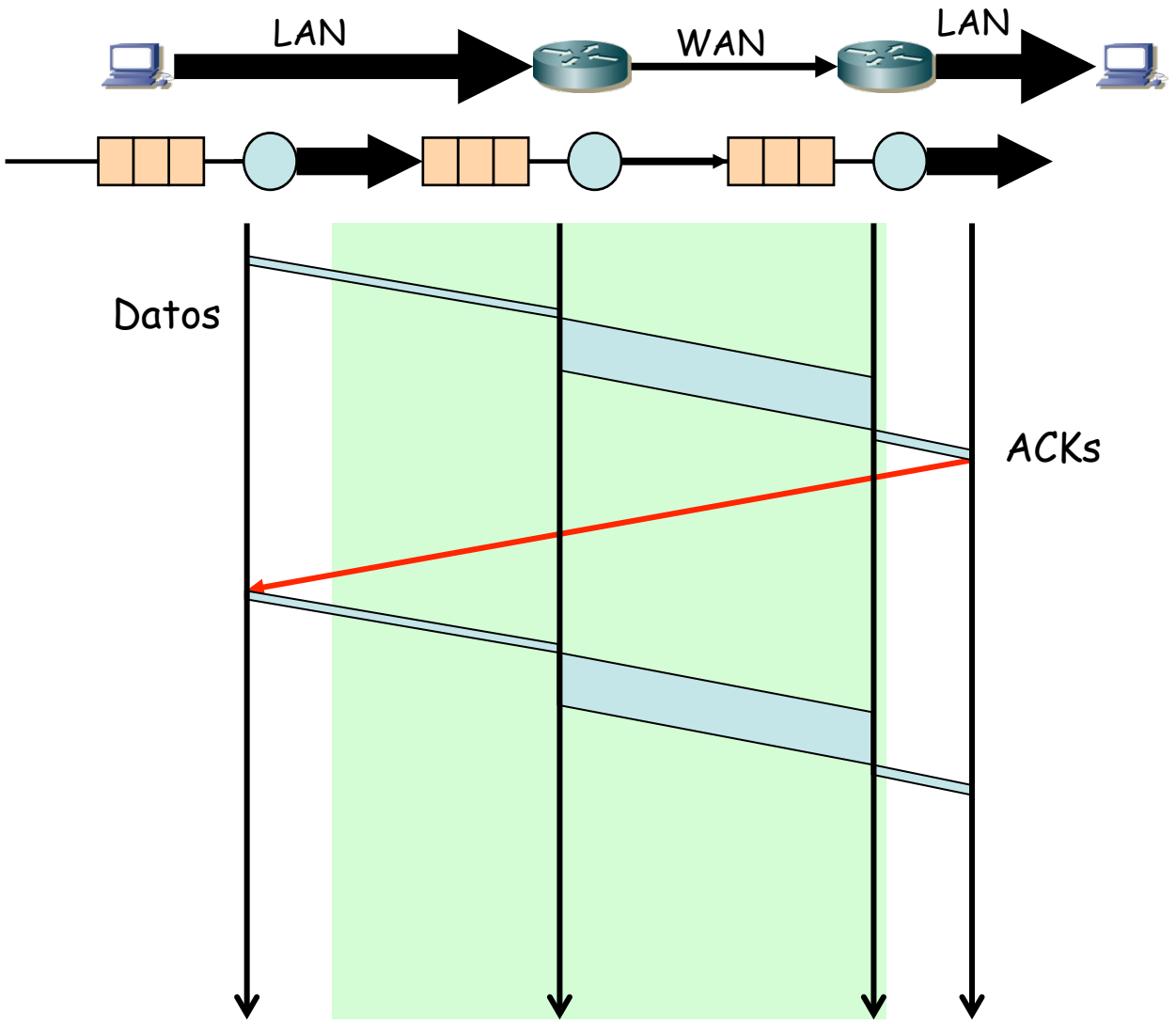
Buffer

- ¿Dónde están los paquetes?
- En buffers o siendo transmitidos



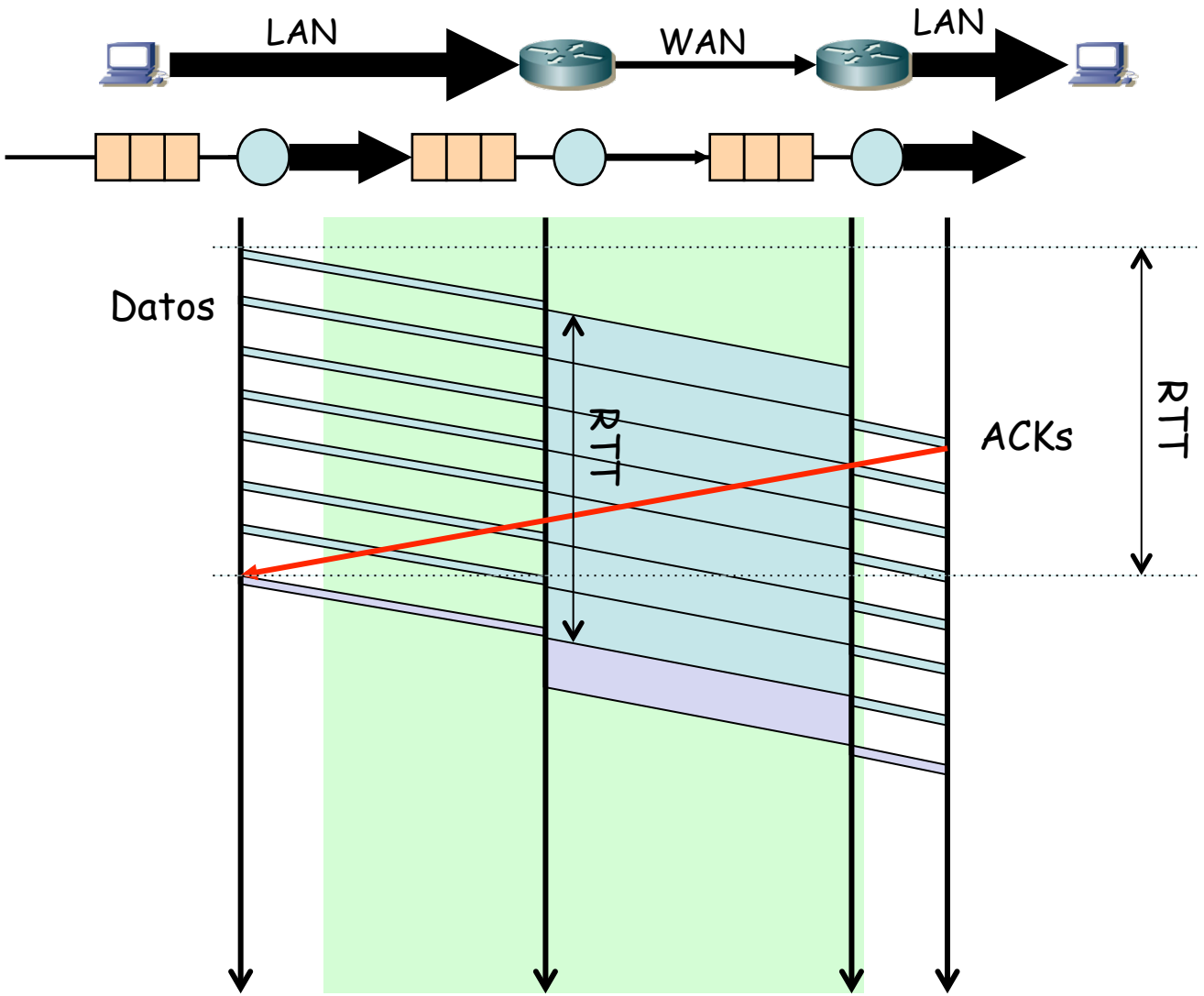


Ejemplo: LAN-WAN-LAN



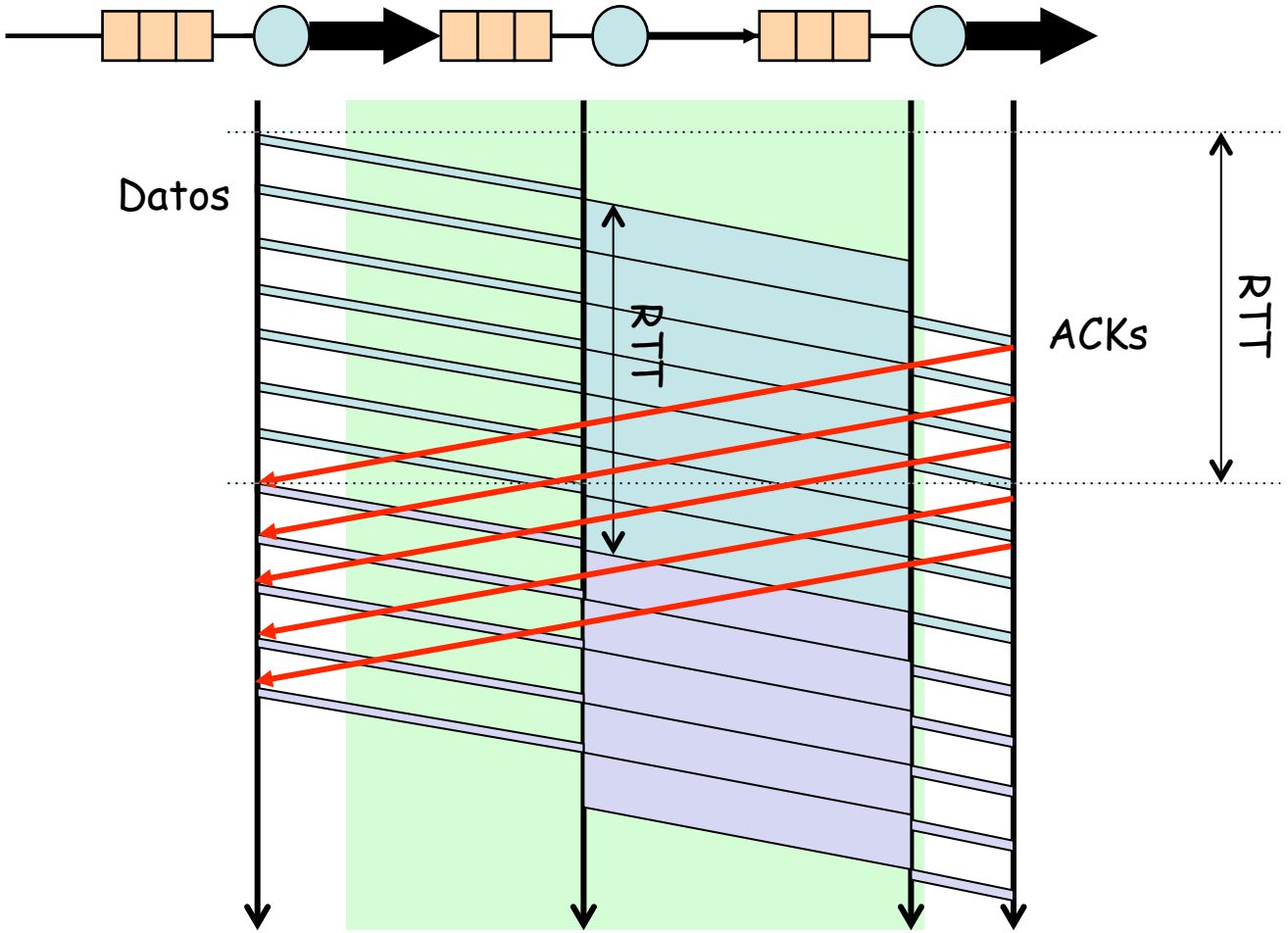
Ejemplo: LAN-WAN-LAN

- Ventana para saturar = RTT x Capacidad_bottleneck



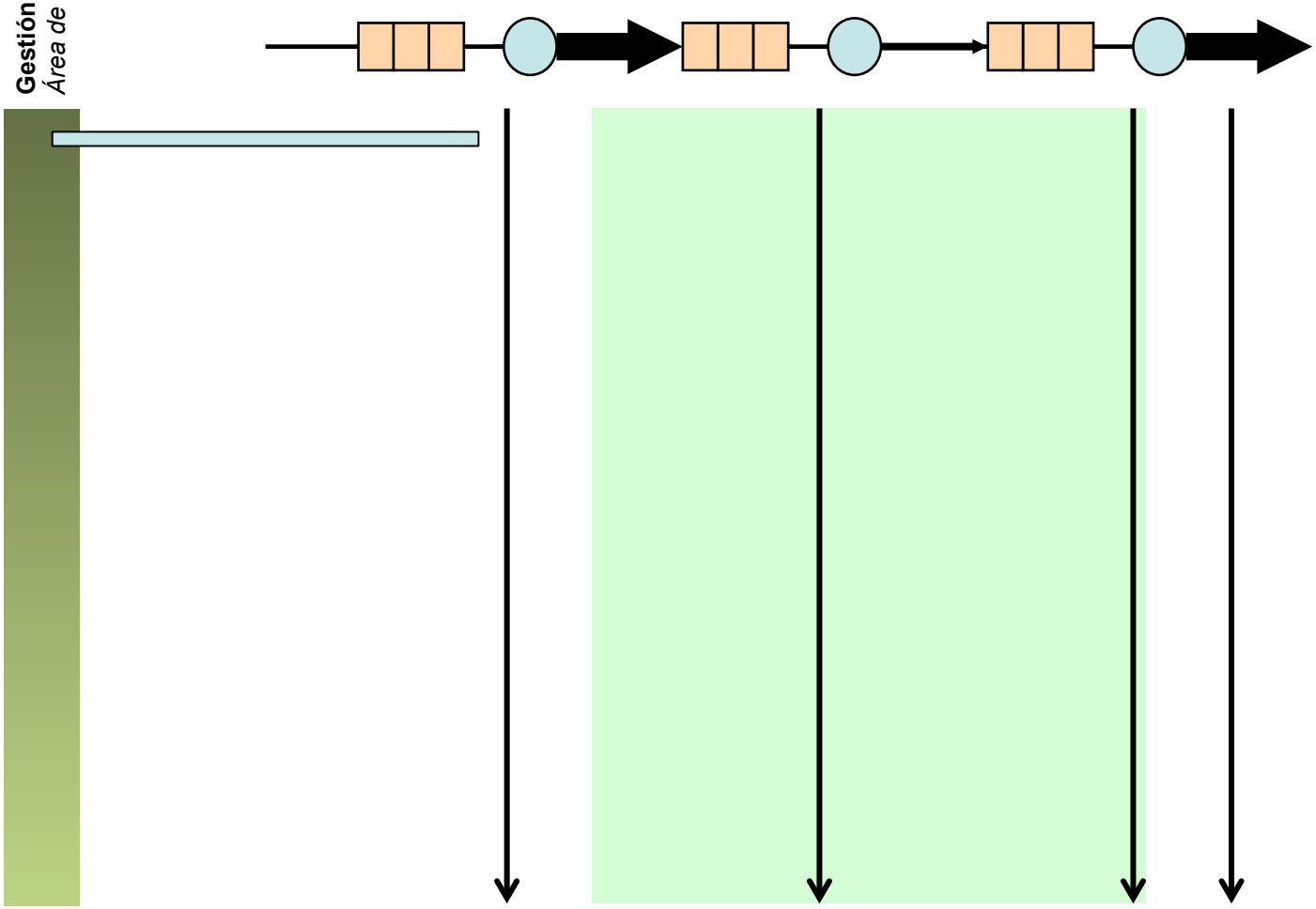
Ejemplo: LAN-WAN-LAN

- Ventana para saturar = RTT x Capacidad_bottleneck
- Y es el *clocking* aportado por los ACKs quien da la regulación



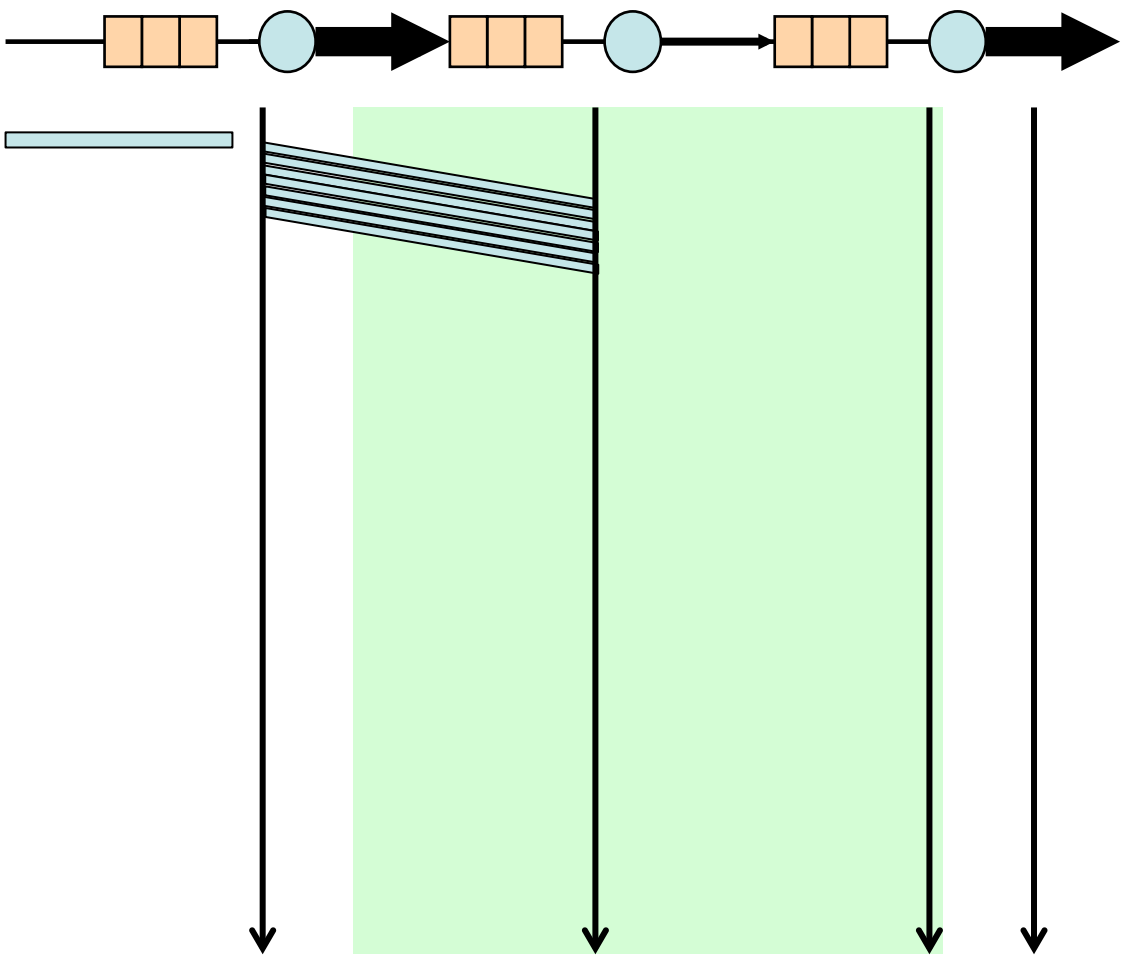
Ejemplo: LAN-WAN-LAN

- Pero si está todo confirmado, y llega un bloque de datos de aplicación
- (…)



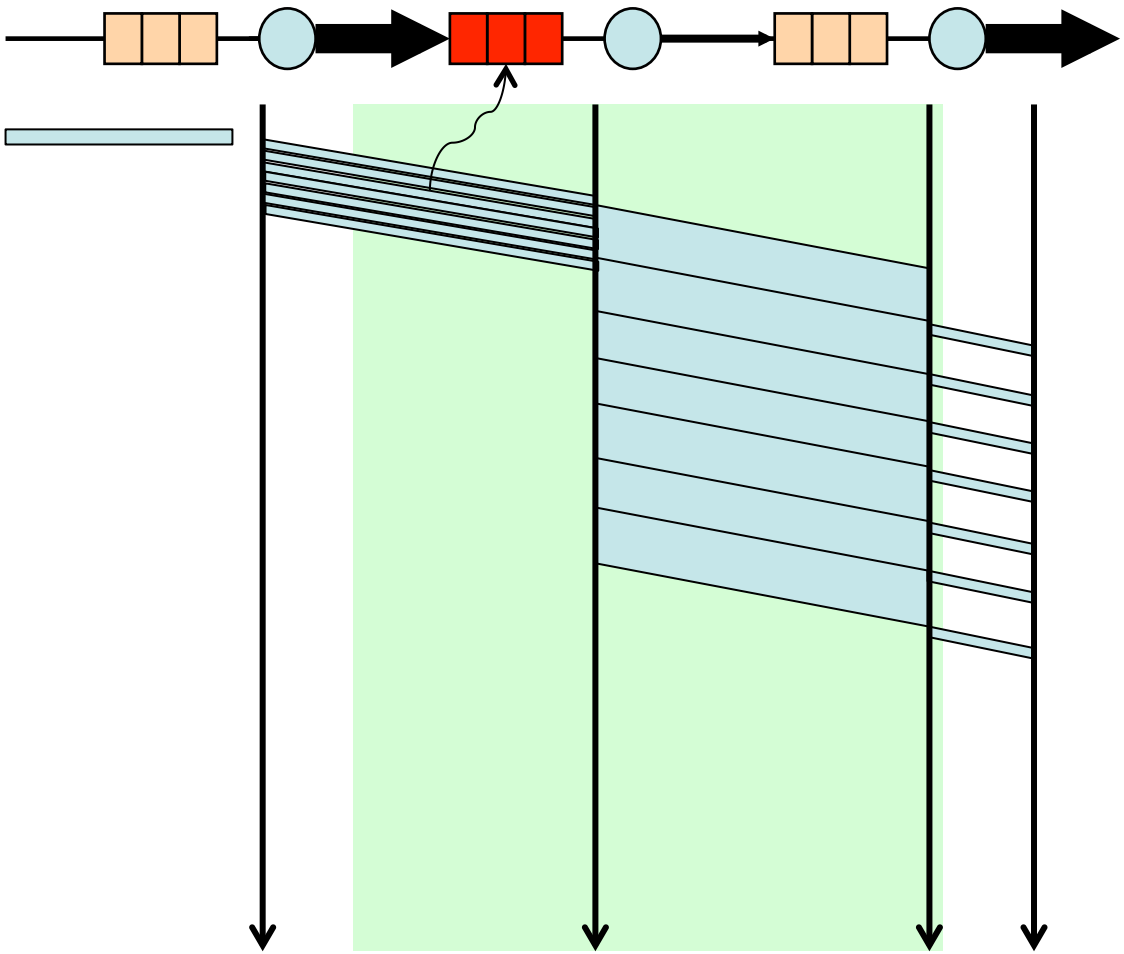
Ejemplo: LAN-WAN-LAN

- Pero si está todo confirmado, y llega un bloque de datos de aplicación
- Se puede enviar toda la ventana como una ráfaga (...)



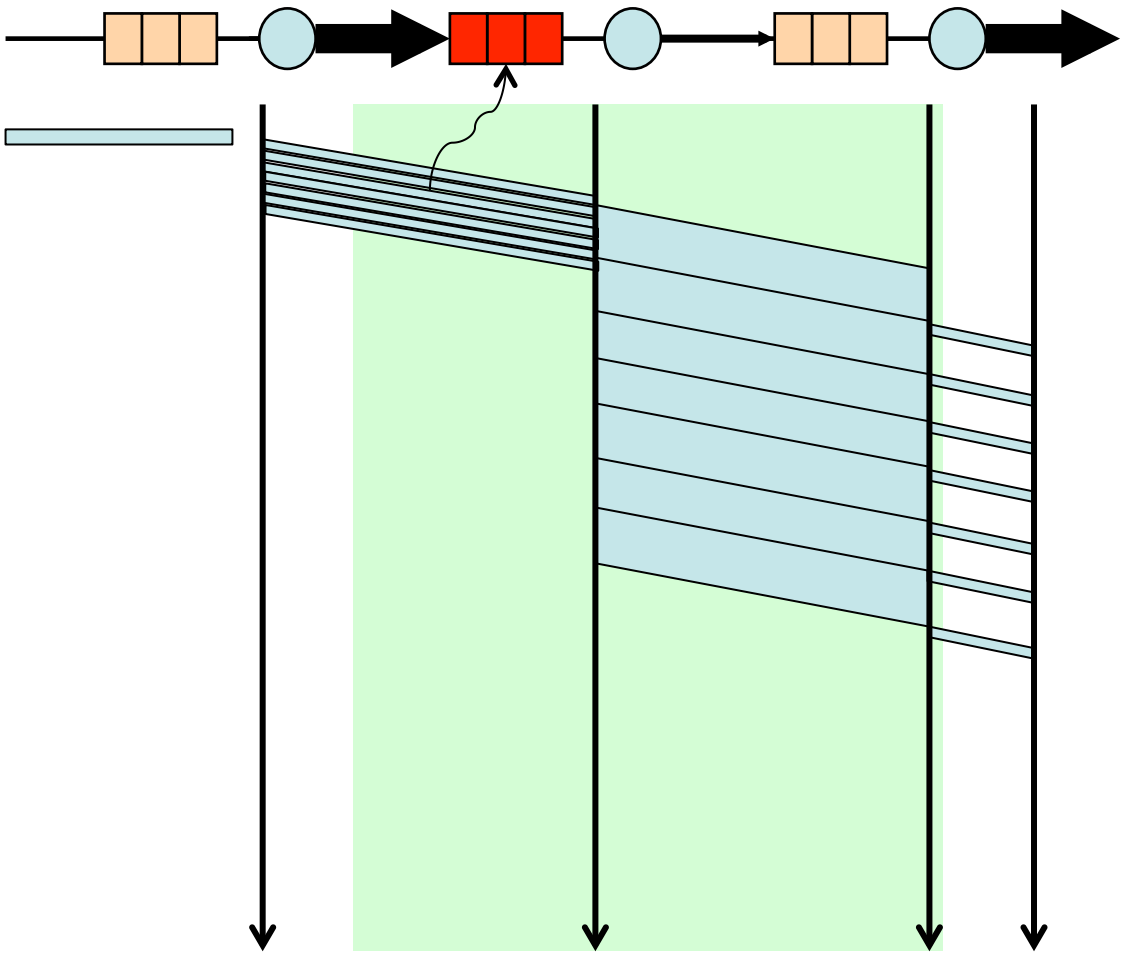
Ejemplo: LAN-WAN-LAN

- Pero si está todo confirmado, y llega un bloque de datos de aplicación
- Se puede enviar toda la ventana como una ráfaga
- Los paquetes quedarán encolados ante el cuello de botella (...)



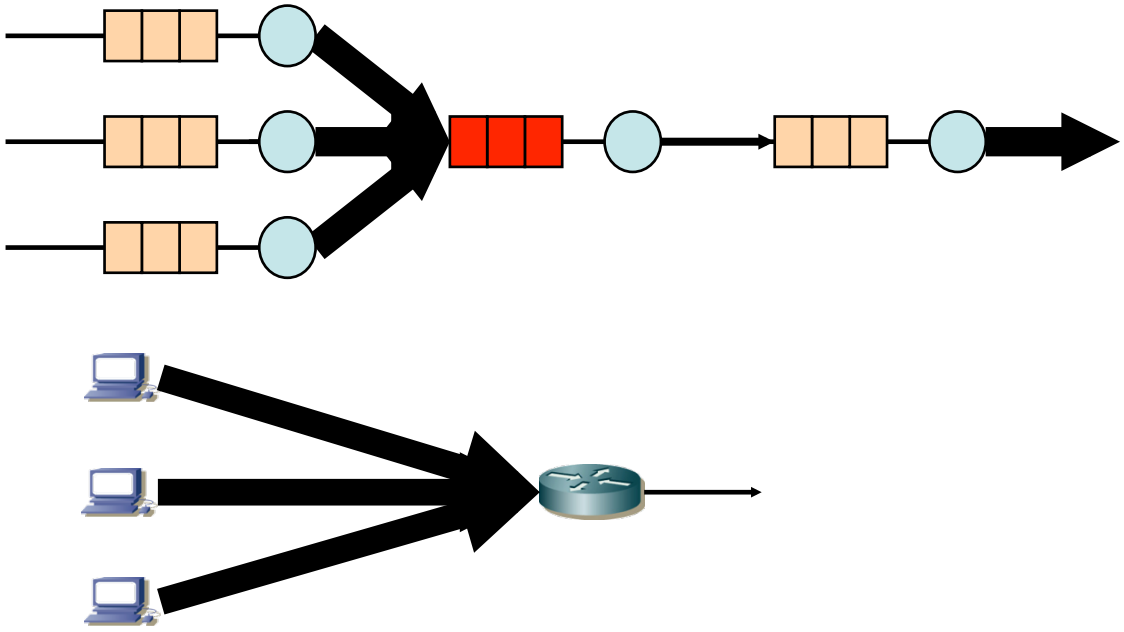
Ejemplo: LAN-WAN-LAN

- De aquí que tradicionalmente se tienda a dimensionar los buffers para que puedan almacenar ese RTTxCapacidad
- Trayecto a 100 Mbps, RTT = 250 ms ➔ más de 3 MBytes (...)



Ejemplo: LAN-WAN-LAN

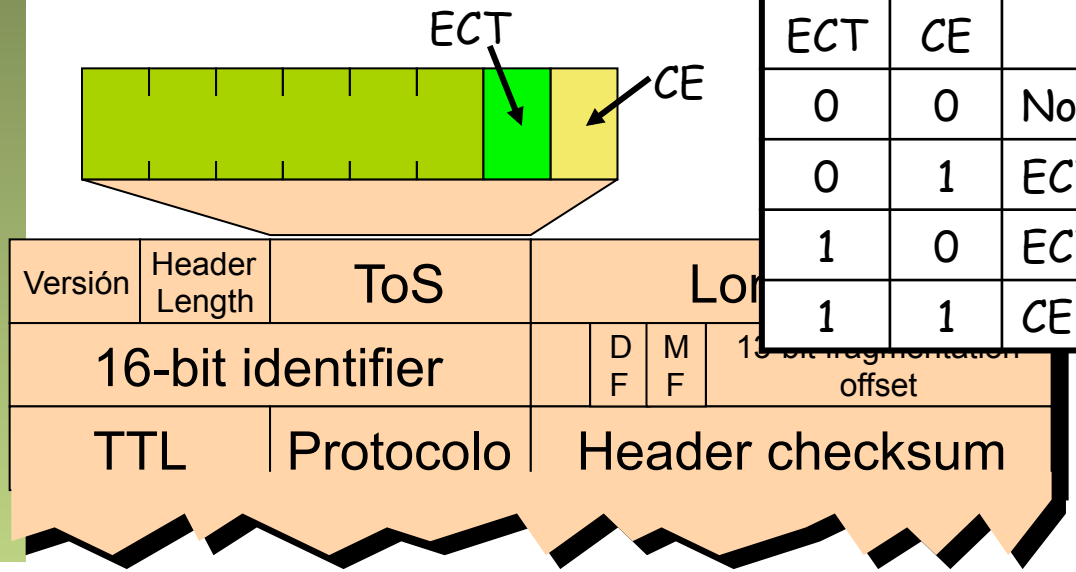
- De aquí que tradicionalmente se tienda a dimensionar los buffers para que puedan almacenar ese RTTxCapacidad
- Trayecto a 100 Mbps, RTT = 250 ms ➔ más de 3 MBytes
- Para un gran número n de flujos TCP (no sincronizados) en realidad los requisitos son más del orden $BWxRTT / \sqrt{n}$
- Hay que tener en cuenta que generalmente el tráfico no viene de un solo interfaz ni es una sola conexión



Realimentación explícita

Realimentación explícita

- Hasta ahora la realimentación sobre congestión era implícita
  - Si detectamos pérdida (timeout) es que debe haber congestión
  - Sale un paquete (llega ACK) es que no debe haberla
- Requiere crear la congestión
- Y que de verdad las pérdidas sean por congestión
- La alternativa explícita es ECN (Explicit Congestion Notification)
- Empleando AQM tipo RED y en lugar de descartar, marcar el paquete

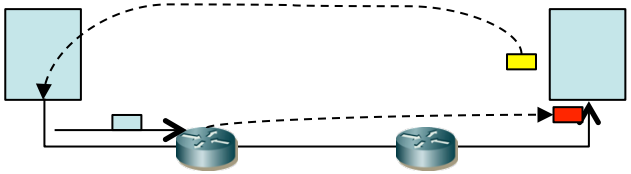


ECN Field			
ECT	CE		
0	0	Not-ECT	Paquete no emplea ECN
0	1	ECT(1)	Extremos "ECN-capable"
1	0	ECT(0)	idem (recomendado)
1	1	CE	Congestion experienced

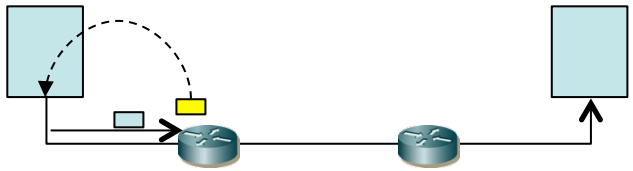
Realimentación explícita

- Con ECN la realimentación la introduce el router
- Pero tiene que llegar al receptor y volver al emisor

CWR	ECE	URG	ACK	PSH	RST	SYN	FIN



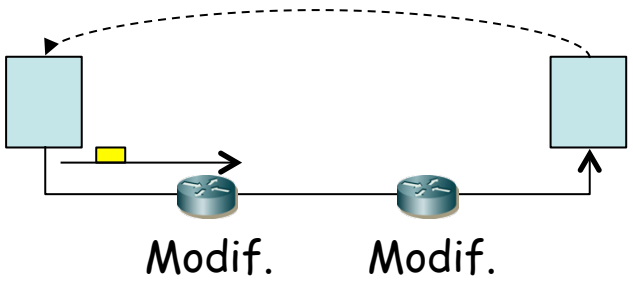
- Otra alternativa es que el router envíe un mensaje al emisor
  - *Choke packet*
  - Permite reacción más rápida



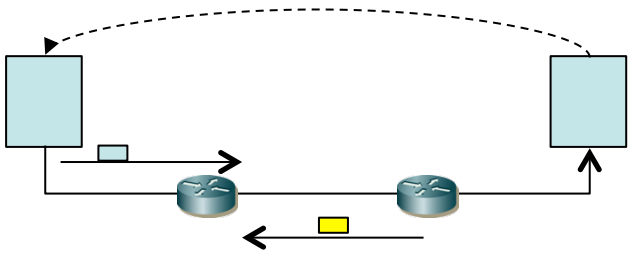
  - Tiene un coste alto para el router que debe crear un paquete nuevo
  - ICMP Source Quench, aunque retirado desde 1995 (RFC 1812 y 6633) y no existe para IPv6
  - Complicado decidir sobre qué flujos actuar
  - Escasa mejora, solo para flujos largos y alto RTT

Realimentación explícita

- Otra alternativa: *explicit rate feedback*
  - La fuente envía paquete o cabecera especial
  - Los conmutadores del camino actualizan información en él
  - Celdas RM en ATM ABR



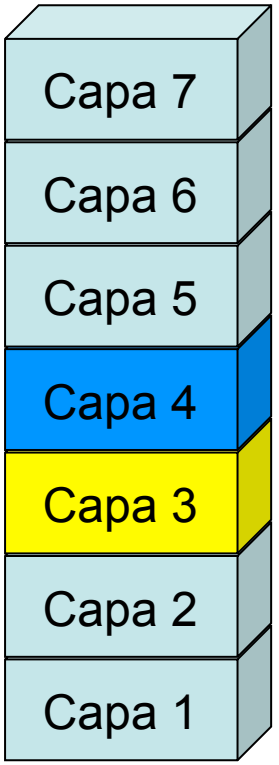
- *Backpressure*
  - *Hop-by-hop congestion control*
  - Cada conmutador del camino envía realimentación al anterior
  - Esquemas hop-by-hop solían plantearse para ofrecer fiabilidad dentro de la red, lo cual se desechó en la Internet (*deadlocks*)



Otros temas

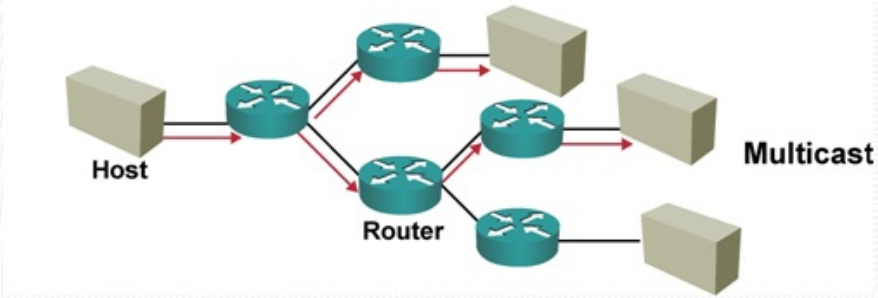
Congestión, TCP/IP y OSI

- OSI habla de flow control en el nivel de red
- Entonces se está refiriendo a lo que llamamos control de congestión pues a lo que protege es a la red
- Tiene sentido en nivel 3 pues ese nivel debe ocultar al nivel 4 lo que sucede en la red
- El que TCP deduzca que hay congestión en base a pérdidas va en contra de los niveles en OSI y da problemas
  - Ej: pérdidas en redes wireless son por interferecia
- Es que TCP/IP es anterior a OSI
- OSI no habla de control de congestión en nivel 4



Otros temas

- ¿Multicast?
  - Realimentación explícita puede inundar a la fuente
  - Diferentes caminos a cada destino, ¿a cuál se adapta?



- ¿Y si hay fuentes egoistas?
  - No aplican control de congestión
  - El resultado es peor para todos
  - ¿Control del tráfico por la red?
  - ¿Que paguen más? (control por el usuario final)