

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Tráfico de datos

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

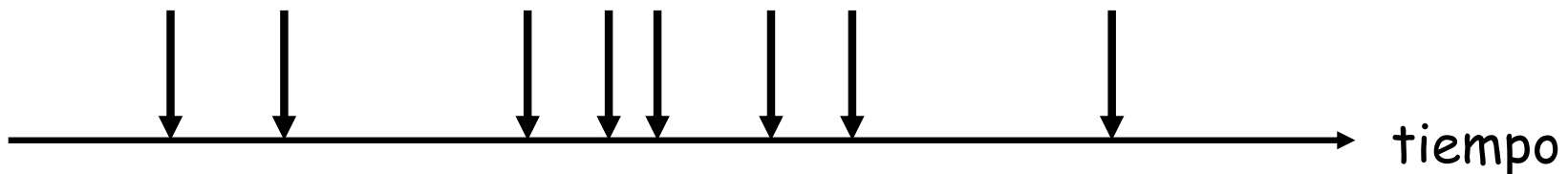
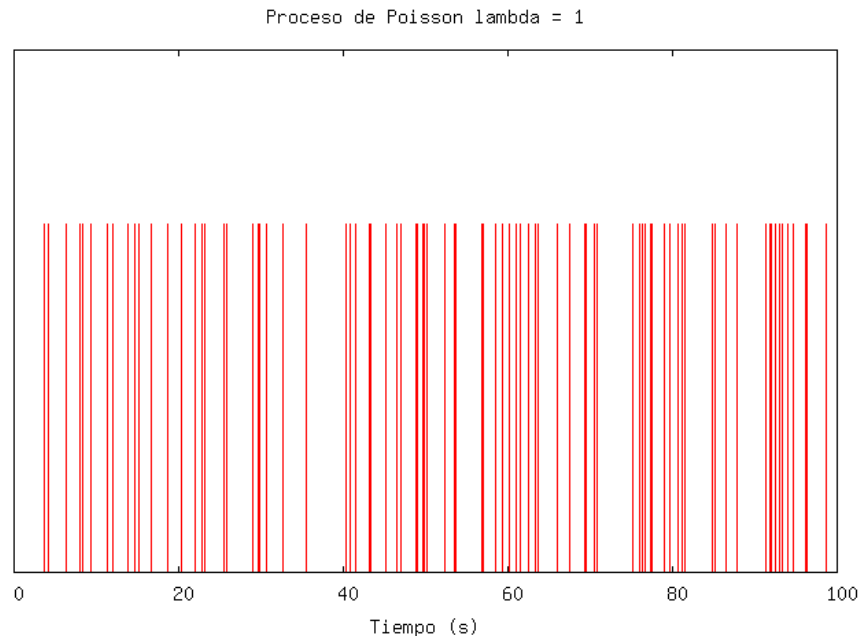
Redes de Nueva Generación
Área de Ingeniería Telemática

Tráfico de datos y Poisson

Desde la telefonía...

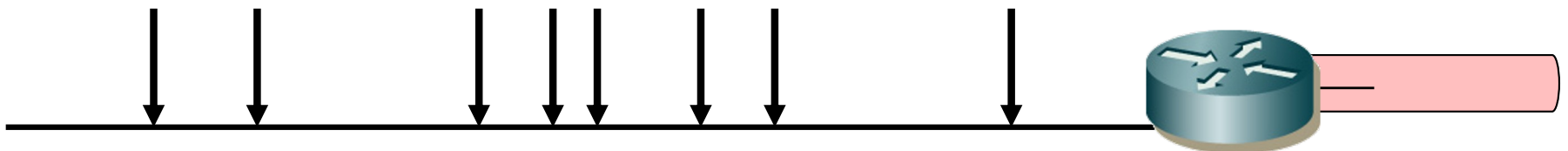
- Empleamos un modelo de llegadas de Poisson
- Modela la distribución del número de llegadas en un intervalo de tiempo
- Es equivalente a tiempos entre llegadas exponenciales i.i.d.
- ¿Es aplicable a tráfico de datos?

$$P[N = k] = \frac{(\lambda \Delta t)^k}{k!} e^{-\lambda \Delta t}$$



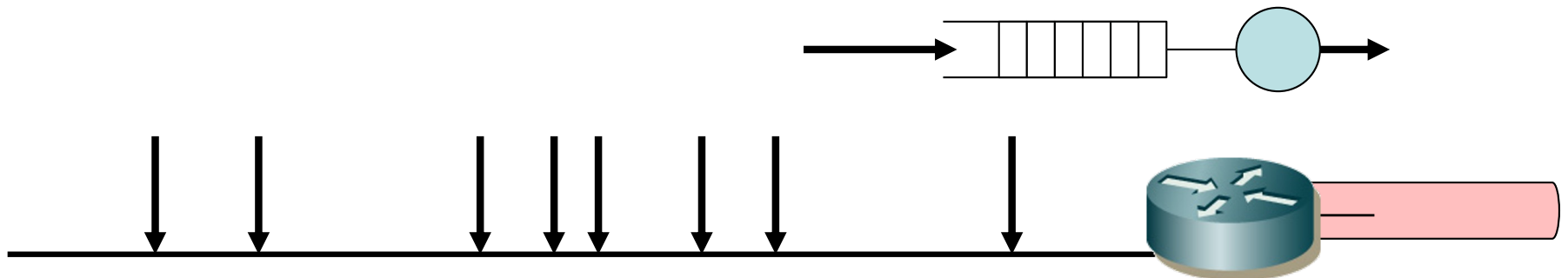
Poisson para datos

- Las llegadas no serían de llamadas sino de paquetes
- La duración es de los paquetes (tiempo de transmisión)
- Solo hay un canal de salida, el enlace de salida
- Ahora no podemos emplear un modelo Erlang-B, ¿por qué?



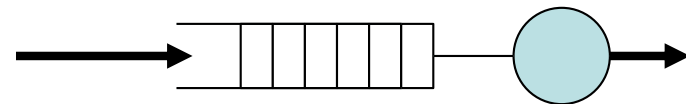
Poisson para datos

- Las llegadas no serían de llamadas sino de paquetes
- La duración es de los paquetes (tiempo de transmisión)
- Solo hay un canal de salida, el enlace de salida
- Ahora no podemos emplear un modelo Erlang-B
- No se descarta una nueva llegada ante que el canal de salida esté ocupado (no *blocked calls cleared*)
- Se encolan
- Teoría de colas



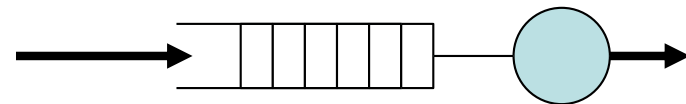
Poisson para datos

- ¿Podemos resolver sistemas de colas con llegadas de Poisson?
- Calcular probabilidades de pérdidas (ante cola llena)
- Calcular distribuciones de tiempo de espera en cola
- Sí, se puede
- ¿Es útil?



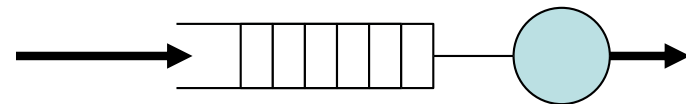
Poisson para datos

- ¿Podemos resolver sistemas de colas con llegadas de Poisson?
- Calcular probabilidades de pérdidas (ante cola llena)
- Calcular distribuciones de tiempo de espera en cola
- Sí, se puede
- ¿Es útil? No
- ¿Por qué?



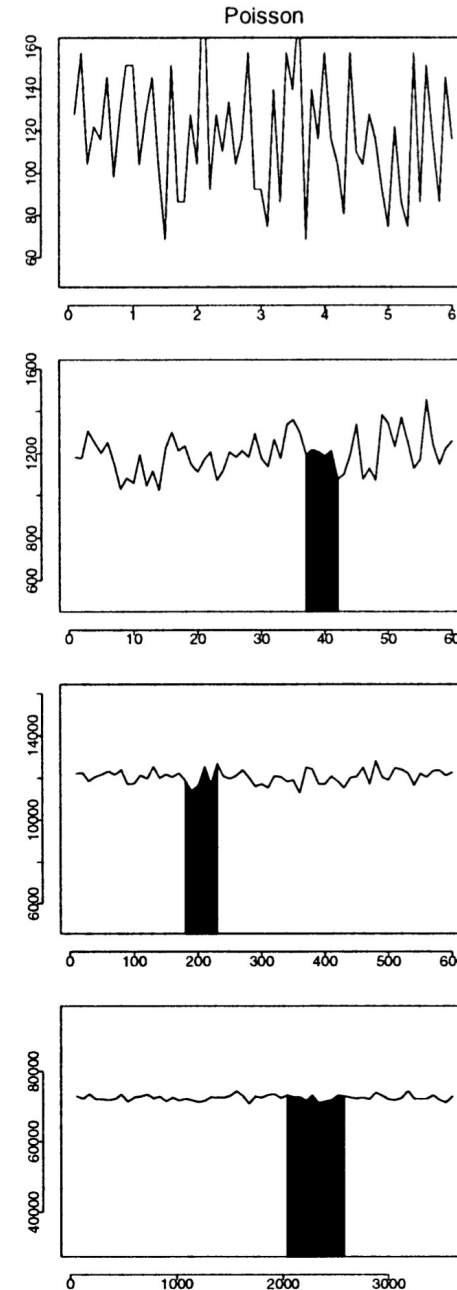
Poisson para datos

- ¿Podemos resolver sistemas de colas con llegadas de Poisson?
- Calcular probabilidades de pérdidas (ante cola llena)
- Calcular distribuciones de tiempo de espera en cola
- Sí, se puede
- ¿Es útil? No
- ¿Por qué?
- Porque el tráfico no es de Poisson



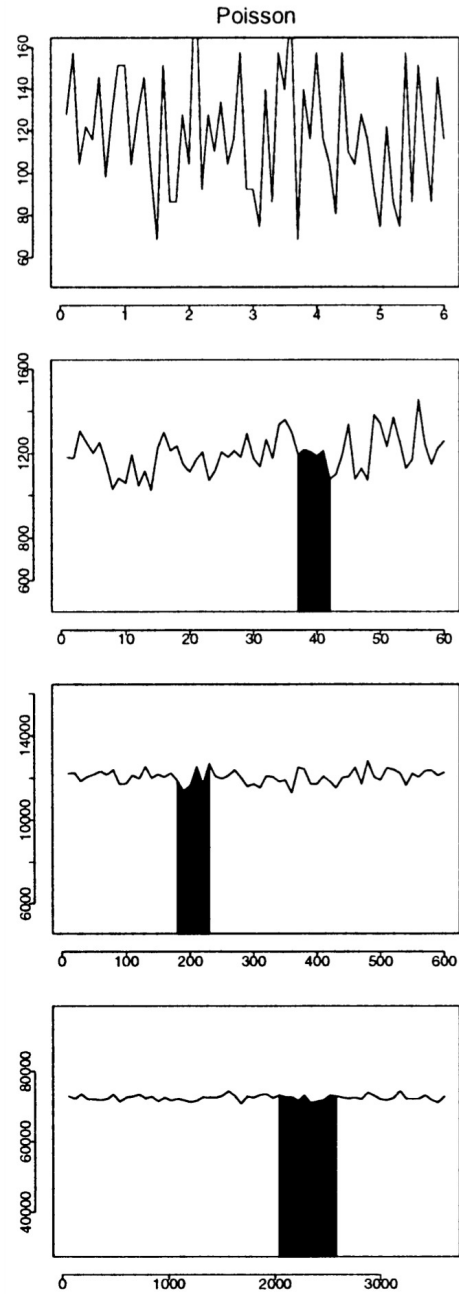
¿Qué tiene de bueno Poisson?

1. Matemáticamente tratable
 - M/M/1 y familia
 - Si fuera de Poisson podríamos dar factores de utilización del 99% con retardos en cola por debajo de 1ms
2. Se suaviza al agregar
 - Dimensionar ligeramente sobre la media

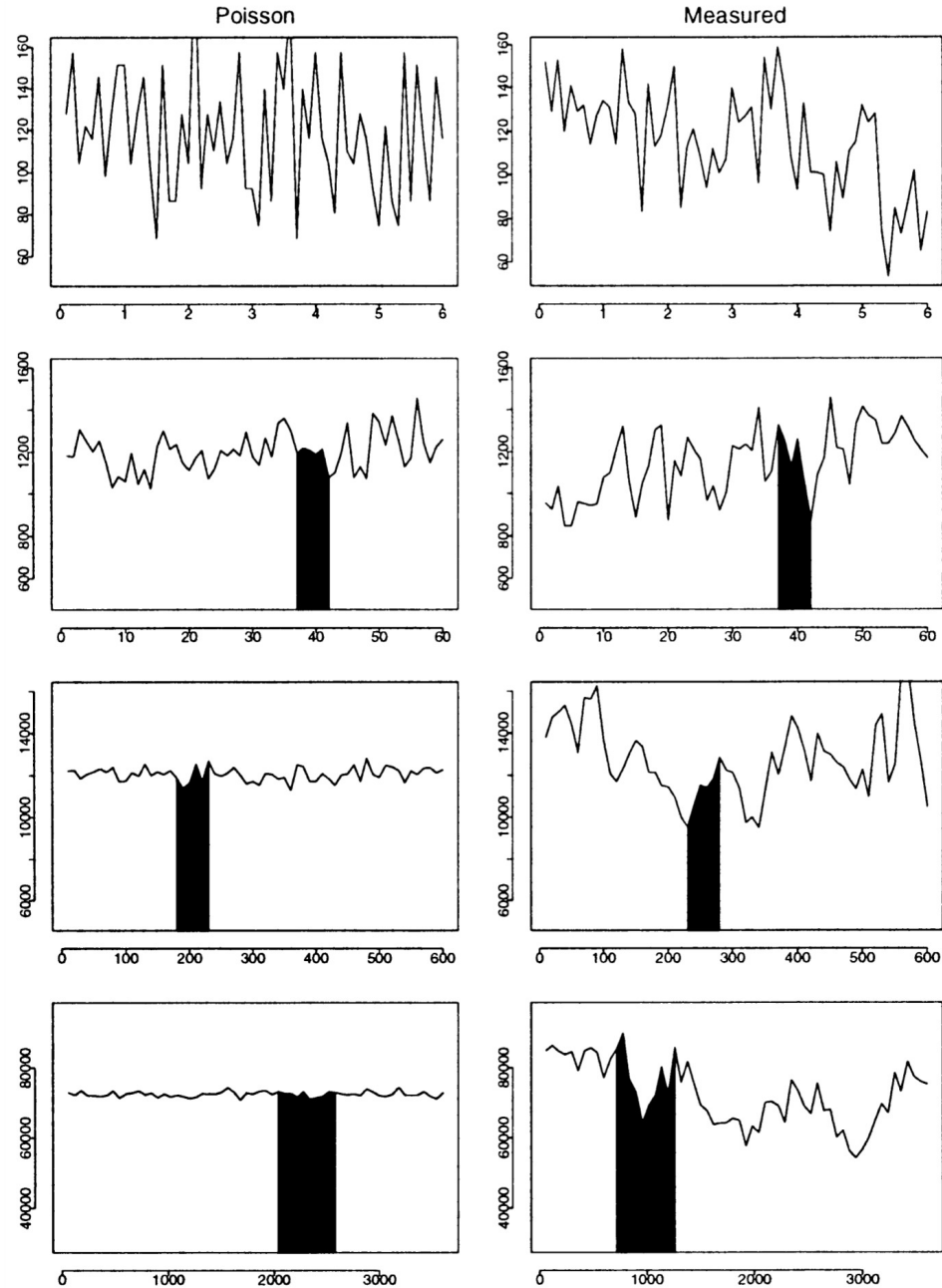


¿Cómo es el tráfico?

¿Cómo es el tráfico?

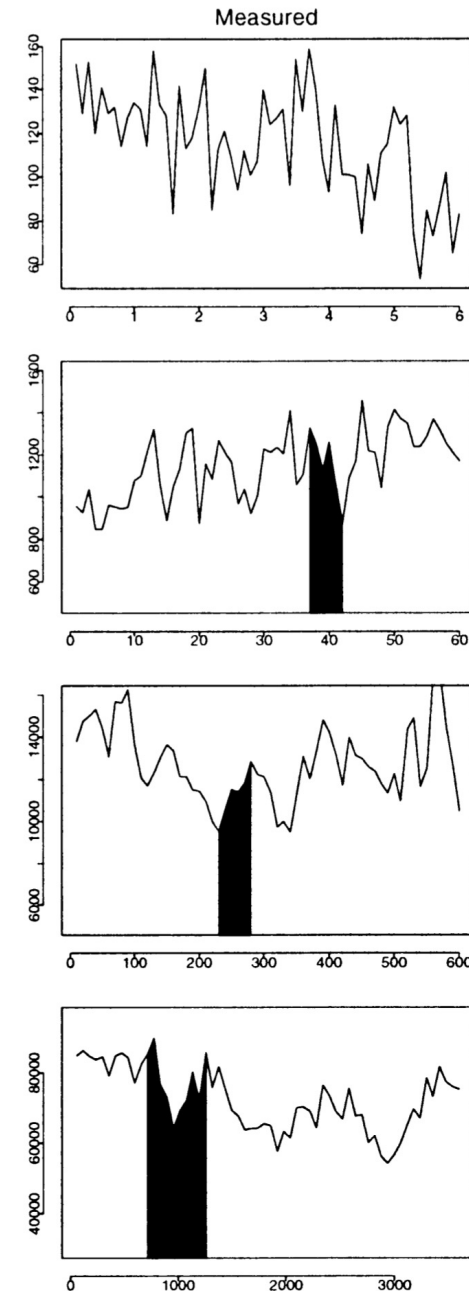


¿Cómo es el tráfico?



¿Cómo es el tráfico?

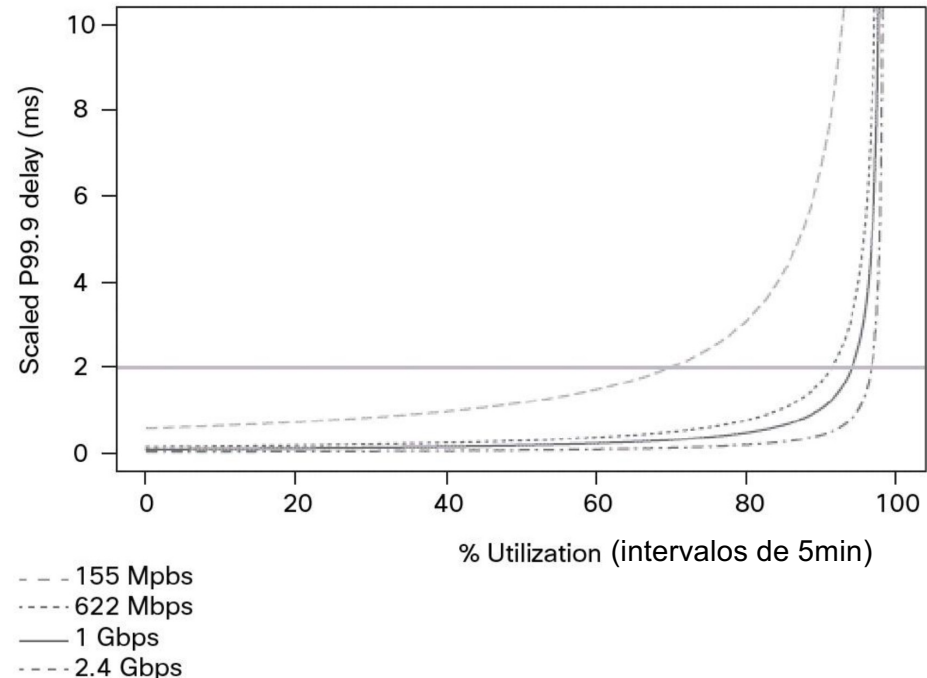
- Mantiene ráfagas (*burstiness*)
- En todas las escalas de agregación
- Con tráfico de Poisson teníamos que considerar muy pequeña variación con alta agregación
- Ahora con el nivel de agregación la variación no se reduce tan rápido
- Para mantener un retardo bajo debemos trabajar en factores de utilización moderados (¡nada del 99%!).
- Modelos auto-similares, fractales
- Discutidos, lo que importan son las escalas bajas, modelos multi-fractales (¡!).
- Aproximaciones numéricas por simulación



Ejemplo de dimensionamiento

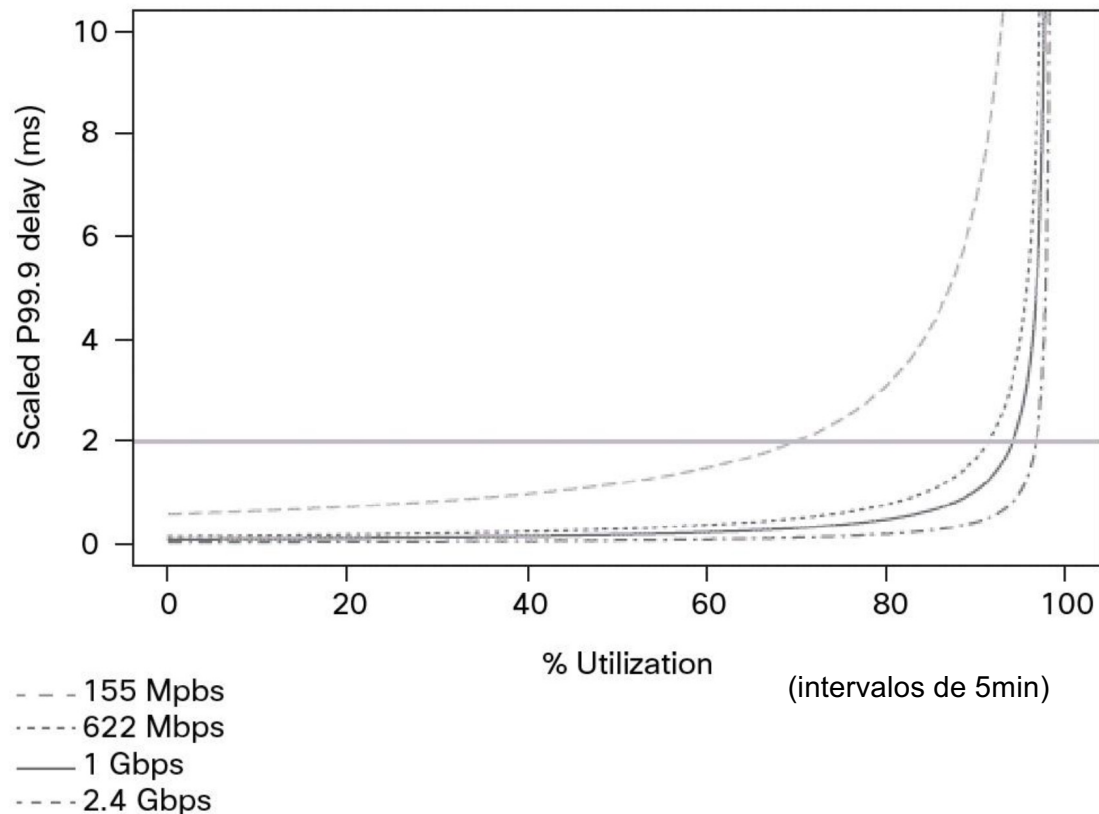
Ejemplo de aproximación

- Thomas Telkamp, “Traffic Characteristics and Network Planning”, presentación en el NANOG 26, 2002
- Intenta mejorar reglas como “deberías aumentar la capacidad del enlace si su utilización llega al 50%”
- Se obtienen (para el mismo factor de utilización) menores retardos cuanto mayor sea la capacidad del enlace (mayor grado de mux.)
- Es decir, aunque no sea tráfico de Poisson sí hay ganancia al agregar, reflejada en menor *overprovisioning* necesario
- Ejemplo (...)



Ejemplo de aproximación

- Ejemplo:
 - Enlace 155 Mbps, objetivo de 2ms de retardo en el 99.9% de paquetes
 - Utilización máxima del 70%
 - Es decir, como mucho $155 \times 0.7 = 108.5$ Mbps en uso
 - Eso es un *overprovisioning* de $1/0.7 = 1.4$, es decir, o 1.4x el tráfico en uso
 - Con un enlace de 2.4Gbps podríamos llegar a un 95% de utilización
 - Eso es solo un 1.05x



upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Tráfico de datos y Poisson

Tráfico de datos en el DC

upna

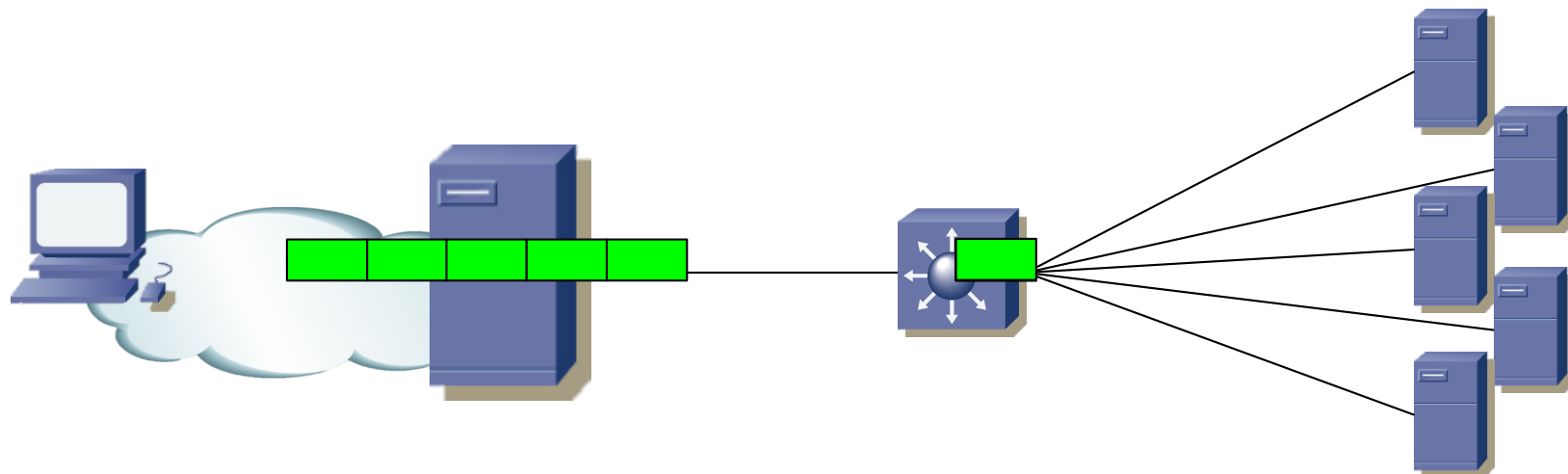
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Incast

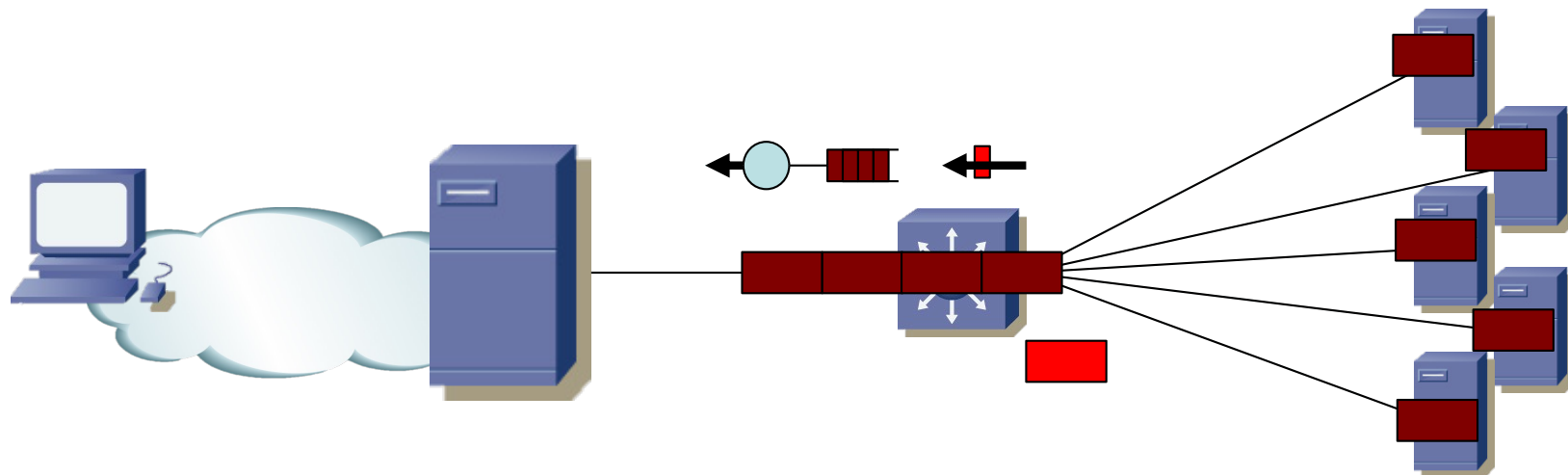
Incast: Fenómeno

- El fenómeno se da en escenarios de aplicaciones con esquema *Partition+Aggregate* y *barrier synchronized*
- Ejemplo:
 - Cliente hace una petición simultáneamente a un gran número de servidores
 - Suelen ser múltiples conexiones TCP (...)



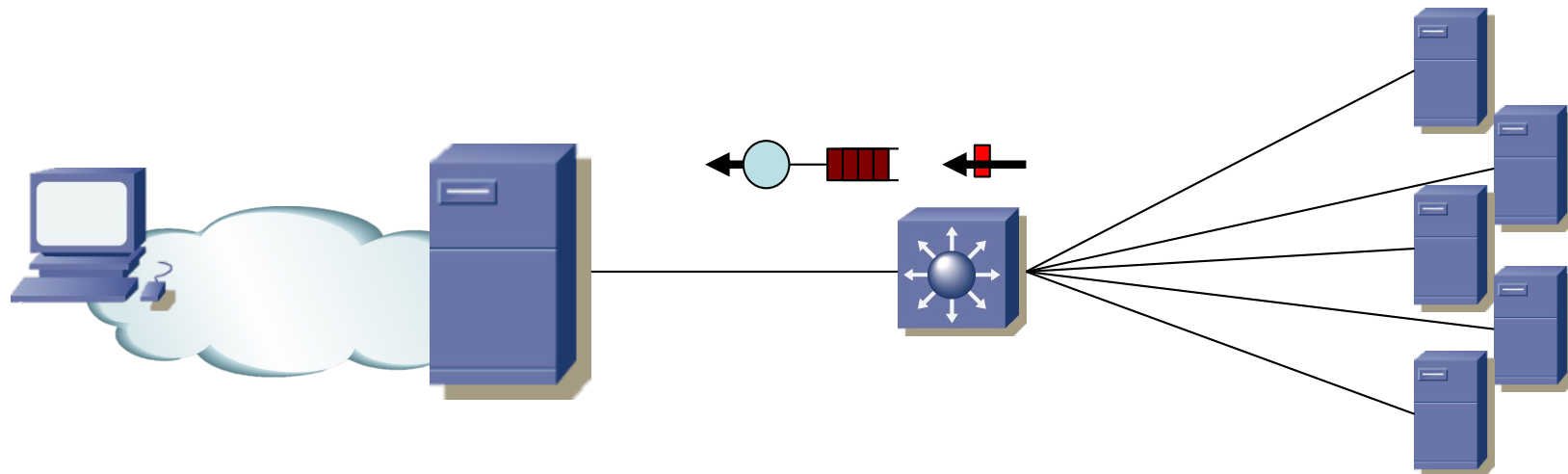
Incast: Fenómeno

- El fenómeno se da en escenarios de aplicaciones con esquema *Partition+Aggregate* y *barrier synchronized*
- Ejemplo:
 - Cliente hace una petición simultáneamente a un gran número de servidores
 - Suelen ser múltiples conexiones TCP (...)
 - Los servidores contestan prácticamente a la vez (...)
 - El tráfico puede saturar el buffer del puerto hacia el cliente (...)



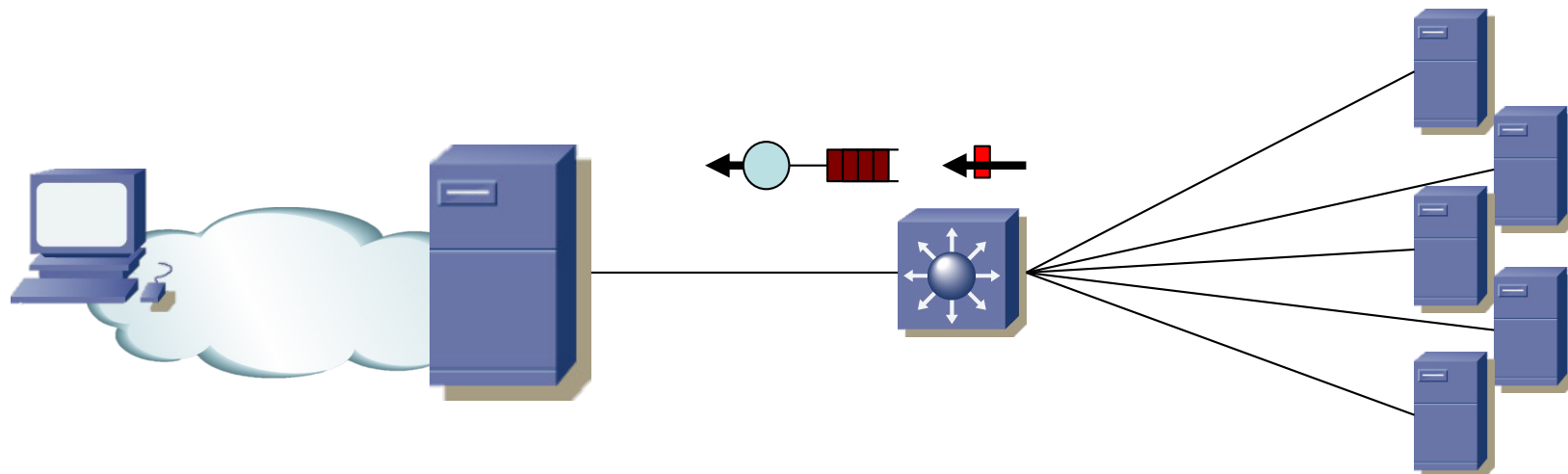
Incast: Fenómeno

- El fenómeno se da en escenarios de aplicaciones con esquema *Partition+Aggregate* y *barrier synchronized*
- Ejemplo:
 - Cliente hace una petición simultáneamente a un gran número de servidores
 - Suelen ser múltiples conexiones TCP (...)
 - Los servidores contestan prácticamente a la vez (...)
 - El tráfico puede saturar el buffer del puerto hacia el cliente (...)
 - Eso son pérdidas de segmentos TCP



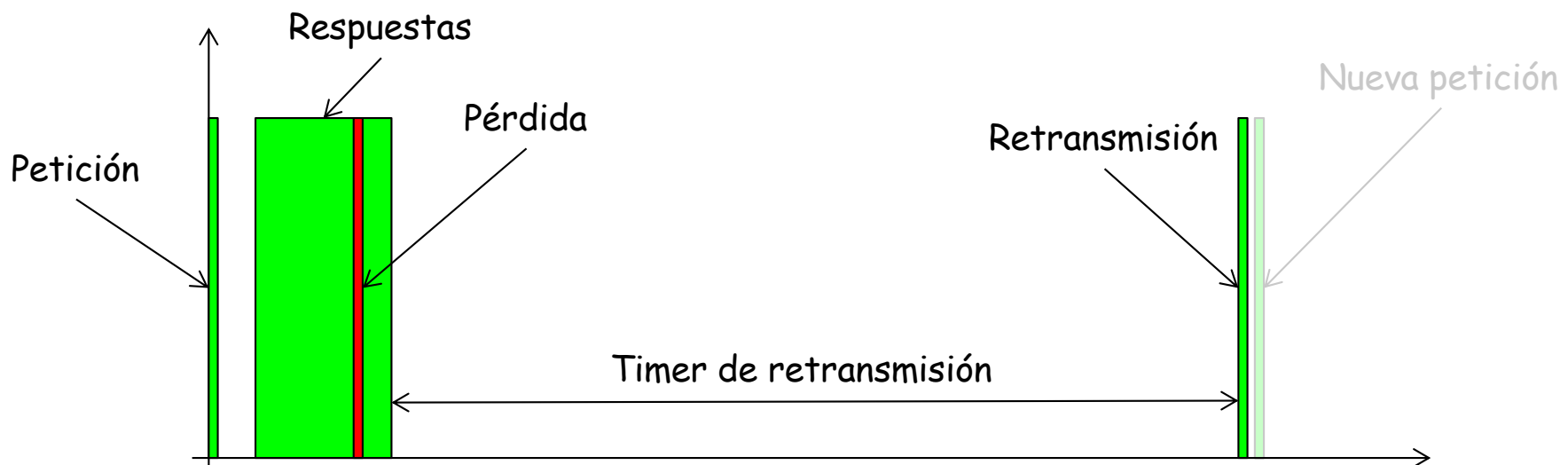
Incast: RTO

- Pérdidas para algunas conexiones
- Las conexiones que sufren pérdidas, si no tienen más datos para enviar, deben recuperarlas mediante timeout
- El cliente no hace nuevas peticiones hasta completar la respuesta de la anterior (*barrier synchronized*)
- El timeout suele estar en el rango de 200-300ms
- Pero la fase de transferencia ha podido durar unos pocos ms (...)



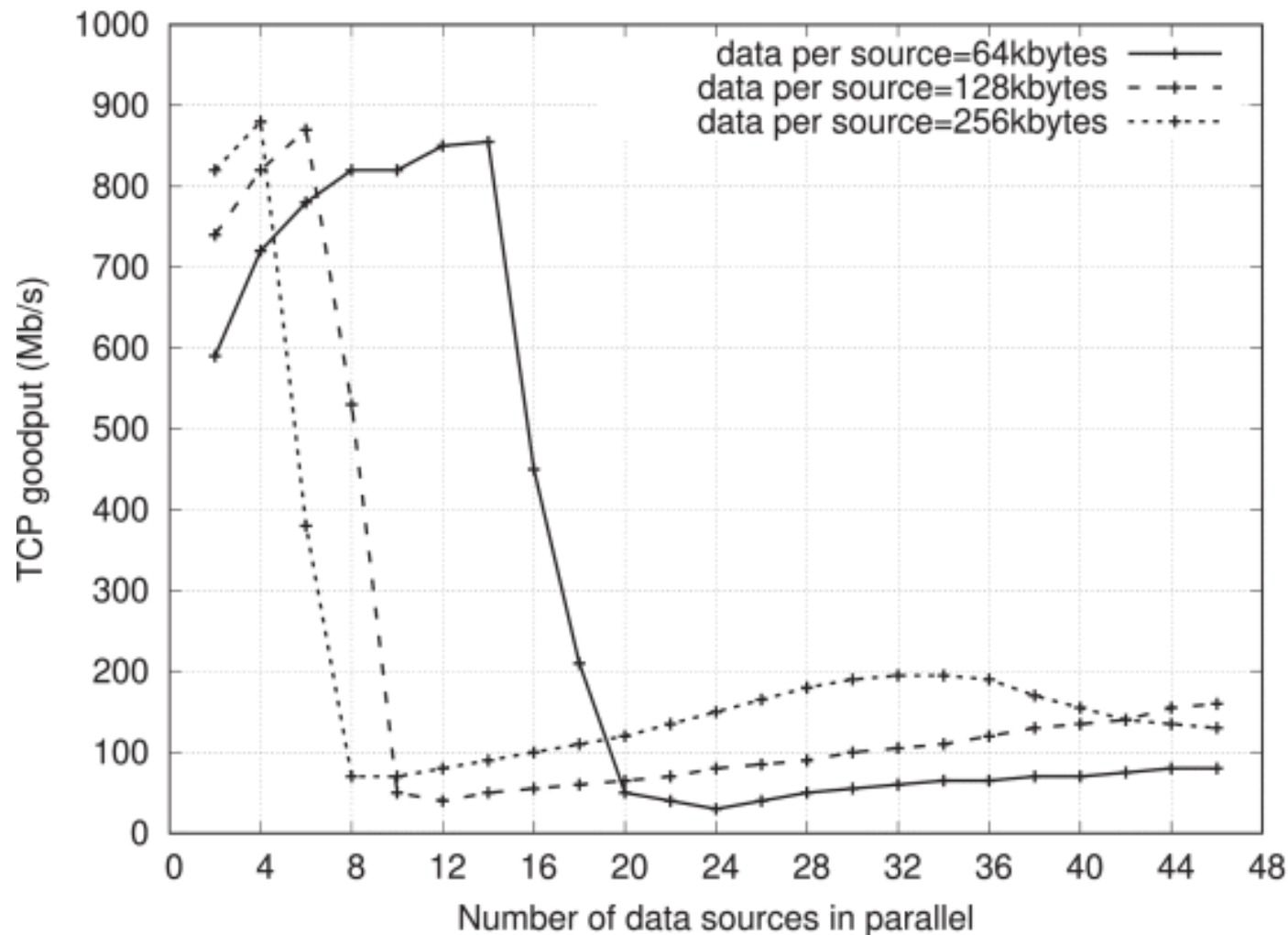
Incast: RTO

- Pérdidas para algunas conexiones
- Las conexiones que sufren pérdidas, si no tienen más datos para enviar, deben recuperarlas mediante timeout
- El cliente no hace nuevas peticiones hasta completar la respuesta de la anterior (*barrier synchronized*)
- El timeout suele estar en el rango de 200-300ms
- Pero la fase de transferencia ha podido durar unos pocos ms
- Eso quiere decir que gran parte del tiempo es de inactividad
- El cliente puede poner un deadline y construir sus respuesta con información incompleta



Incast: Consecuencias

- El resultado es una gran caída en el goodput alcanzado



Incast: Causas

- Buffers pequeños en los conmutadores
- Ejemplo: Cisco Nexus 3048
 - 48x 10/100/1000Mbps + 4x 1/10GbE
 - 176Gbps switching capacity, 132Mpps
 - 128K MAC addresses, 4096 VLANs, 64 instancias MSTP
 - 16K IPv4 prefixes
 - (...)



Incast: Causas

- Buffers pequeños en los conmutadores
- Ejemplo: Cisco Nexus 3048
 - 48x 10/100/1000Mbps + 4x 1/10GbE
 - 176Gbps switching capacity, 132Mpps
 - 128K MAC addresses, 4096 VLANs, 64 instancias MSTP
 - 16K IPv4 prefixes
 - Pero solo 9MB para buffers, compartida entre los puertos
 - Eso son unos 170KB por puerto
 - ¿Por qué? SRAM on-chip



Incast: Causas

- Ejemplo: Arista 7050SX-96
 - 48x 1/10GbE + 12x 40GbE
 - 2.56Tbps, 1440Mpps, latencia 550ns (cut-through)
 - 288K MAC entries, 144K IPv4 routes
 - 12Mbytes dynamic buffer (200Kbytes/puerto)



Incast: Causas

- RTO con valor mínimo en 200-300ms
- De hecho el mínimo en la RFC es 1s
- ¿Por qué? (...)

Incast: Causas

- RTO con valor mínimo en 200-300ms
- De hecho el mínimo en la RFC es 1s
- ¿Por qué? TCP está diseñado para la Internet
- ¿RTTs en el datacenter? (...)

Incast: Causas

- RTO con valor mínimo en 200-300ms
- De hecho el mínimo en la RFC es 1s
- ¿Por qué? TCP está diseñado para la Internet
- ¿RTTs en el datacenter? Unos 100 μ s (buffer vacío)
- ¿Cómo? 1518bytes a 1Gbps son 12 μ s, eso 4 veces (1 solo salto por conmutador) son ya unos 50 μ s
- ¿Y con buffer lleno? (...)

Incast: Causas

- RTO con valor mínimo en 200-300ms
- De hecho el mínimo en la RFC es 1s
- ¿Por qué? TCP está diseñado para la Internet
- ¿RTTs en el datacenter? Unos 100µs (buffer vacío)
- ¿Cómo? 1518bytes a 1Gbps son 12µs, eso 4 veces (1 solo salto por conmutador) son ya unos 50µs
- ¿Y con buffer lleno? Depende de la “profundidad” del buffer
 - Ejemplo: 100Kbytes por puerto
 - Suponiendo solo el puerto hacia el cliente saturado
 - RTT de menos de 1ms ($100 \times 1024 \times 8 / 1000000 = 0.8192$ ms)

Incast: Soluciones

- Aumentar el buffer del conmutador
 - Mayor coste
 - Ejemplo: Arista 7280SE-64
 - 48x 1/10GbE + 4x 10/40GbE, 1.44 Tbps, 900Mpps
 - 9GB packet buffer (DRAM, 170MB/puerto)
 - Ojo, ¿quieres 9GBytes en un puerto? Si se llenan dan un retardo en cola (con puerto de 10Gbps) de... ¡ 7 segs !
 - ¿Queremos buffers grandes o pequeños? Si queréis entrar en la discusión:
 - <http://miercom.com/pdf/reports/20160210.pdf>
 - <https://www.arista.com/assets/data/pdf/Whitepapers/BigDataBigBuffers-WP.pdf>
 - <http://video.cisco.com/detail/videos/aci-%E2%80%93-application-centric-infrastructure/video/4796103210001/switch-buffer-requirements-in-networking> (“We do a better job at dropping packets than our competitors” 😊)

Incast: Soluciones

- Mejorar la gestión de los buffers
 - No suelen estar congestionados todos los puertos
 - Repartos dinámicos de memoria
 - <https://docs.broadcom.com/docs/12358325>
 - <http://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-c11-738488.pdf>
 - <http://video.cisco.com/detail/videos/data-center-virtualization/video/4796103209001/nexus-9k-buffering-for-cloud-scale-deployment?autoStart=true>

Incast: Soluciones

- Reducir el RTO mínimo
 - Requiere timers de alta resolución
 - Problemático en entorno WAN, puede llevar a retransmisiones espúreas
- Reaccionar ante la congestión en el nivel de enlace (ej: QCN)
- Reducir el tamaño de las respuestas
 - A nivel de aplicación (también introducir jitter entre ellas)
 - Microsoft reporta¹ casos de Incast con respuestas de 2KBytes
 - Estos flujos suelen convivir con *long-lived flows*

¹M. Alizadeh, A. Greenberg, D.A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta y M. Sridharan (Microsoft Research y Stanford University), "Data Center TCP (DCTCP)", Proceedings of the ACM SIGCOMM 2010 conference

Incast: Soluciones

- Modificar el control de congestión de TCP
 - DCTCP (Data Center TCP)
 - Requiere ECN en los switches
 - Las fuentes reaccionan a las notificaciones de ECN en proporción a la cantidad de ellas recibidas
 - <https://www.microsoft.com/en-us/research/publication/data-center-tcp-dctcp/>
 - ICTCP (Incast congestion Control for TCP)
 - Emplean el anuncio de ventana de control de flujo del receptor para implementar control de congestión
 - El receptor controlar cuánto puede enviar el emisor
 - Con ello pretenden evitar las pérdidas
 - <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/ictcp.pdf>
 - Otros: IA-TCP, D²TCP, TCP-FITDC, TDCTCP, etc.

Incast: Afectados

- Aplicaciones para las que sucede
 - MapReduce
 - Cluster storage
 - Web search (basado en partition+aggregate)
 - Composición de contenido en redes sociales
- En cualquier caso, depende del escenario que se produzca el fenómeno o no
 - Número de emisores
 - Tamaño de bloque que envían
 - Sincronización en el envío
 - Tamaño y gestión del buffer del conmutador
 - Etc.

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Incast

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

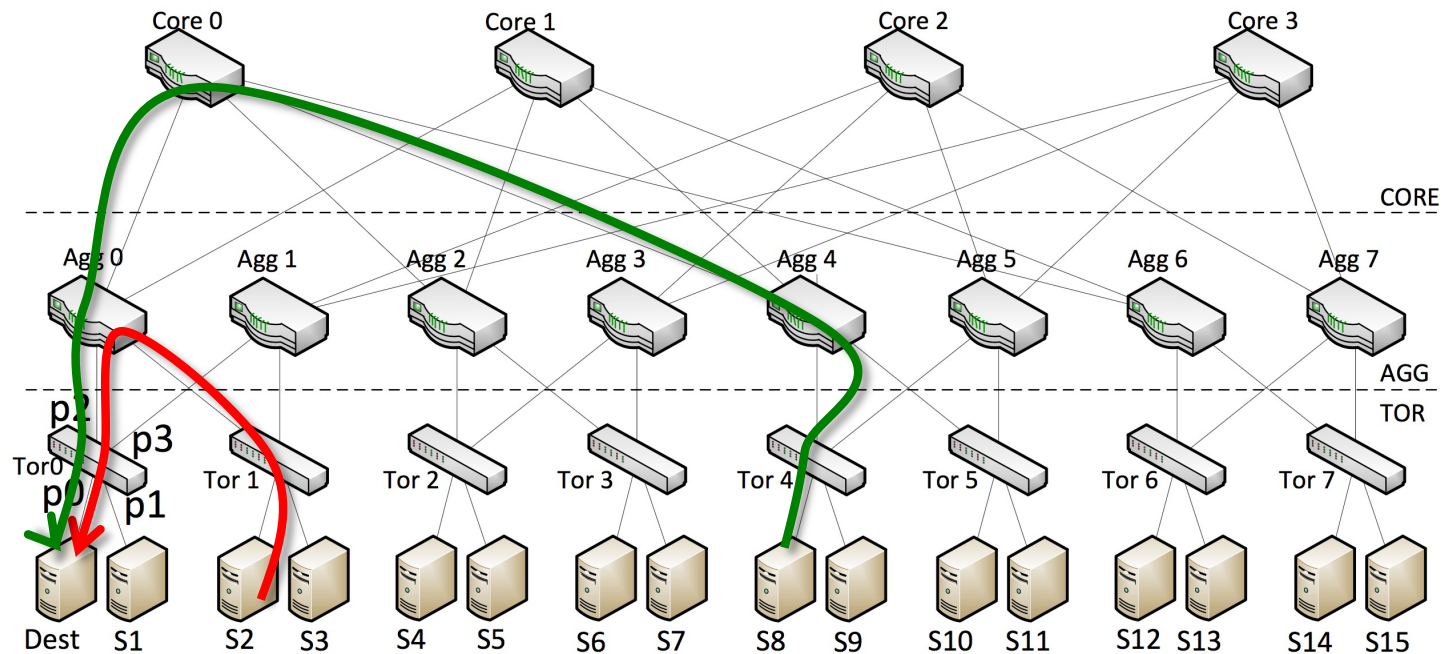


Outcast



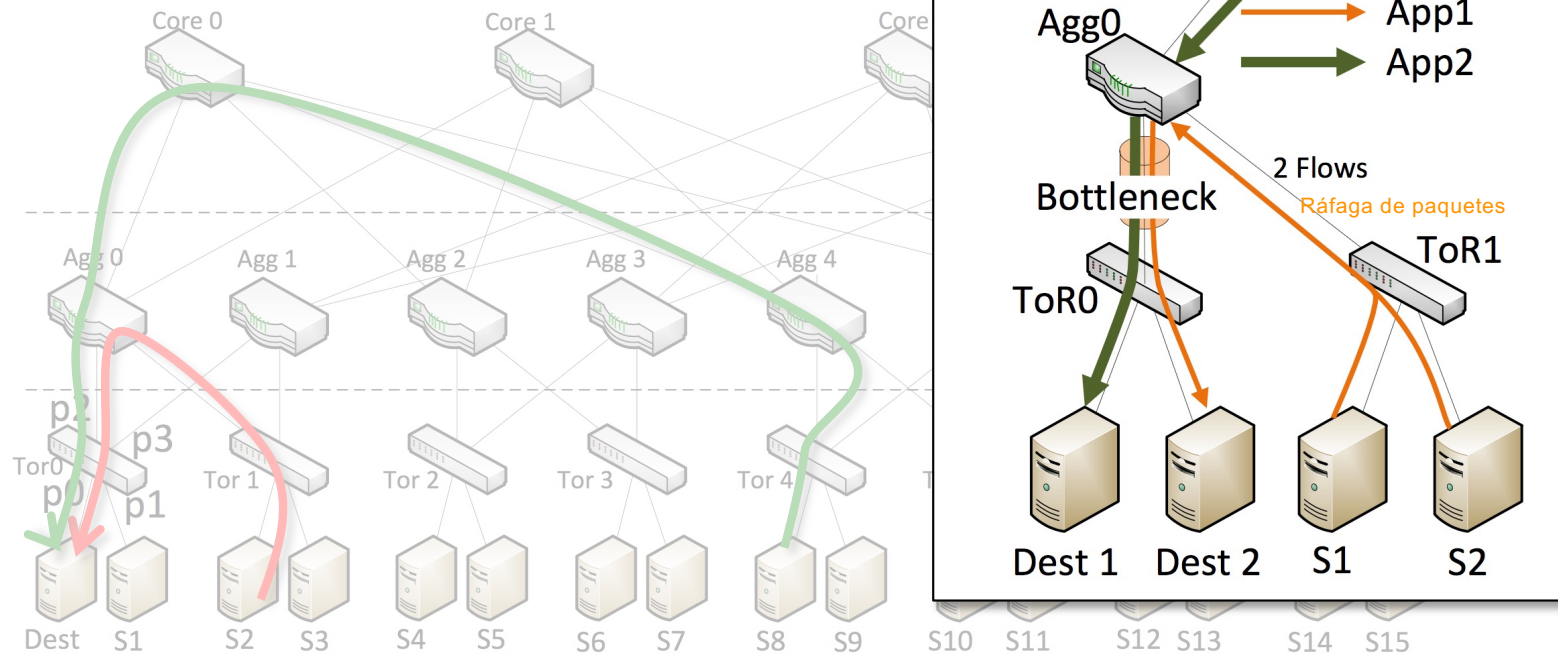
TCP Outcast

- Partition+Aggregate, drop-tail
- Muchos más flujos de servidores alejados que de cercanos, llegan por puertos diferentes (...)
- (...)



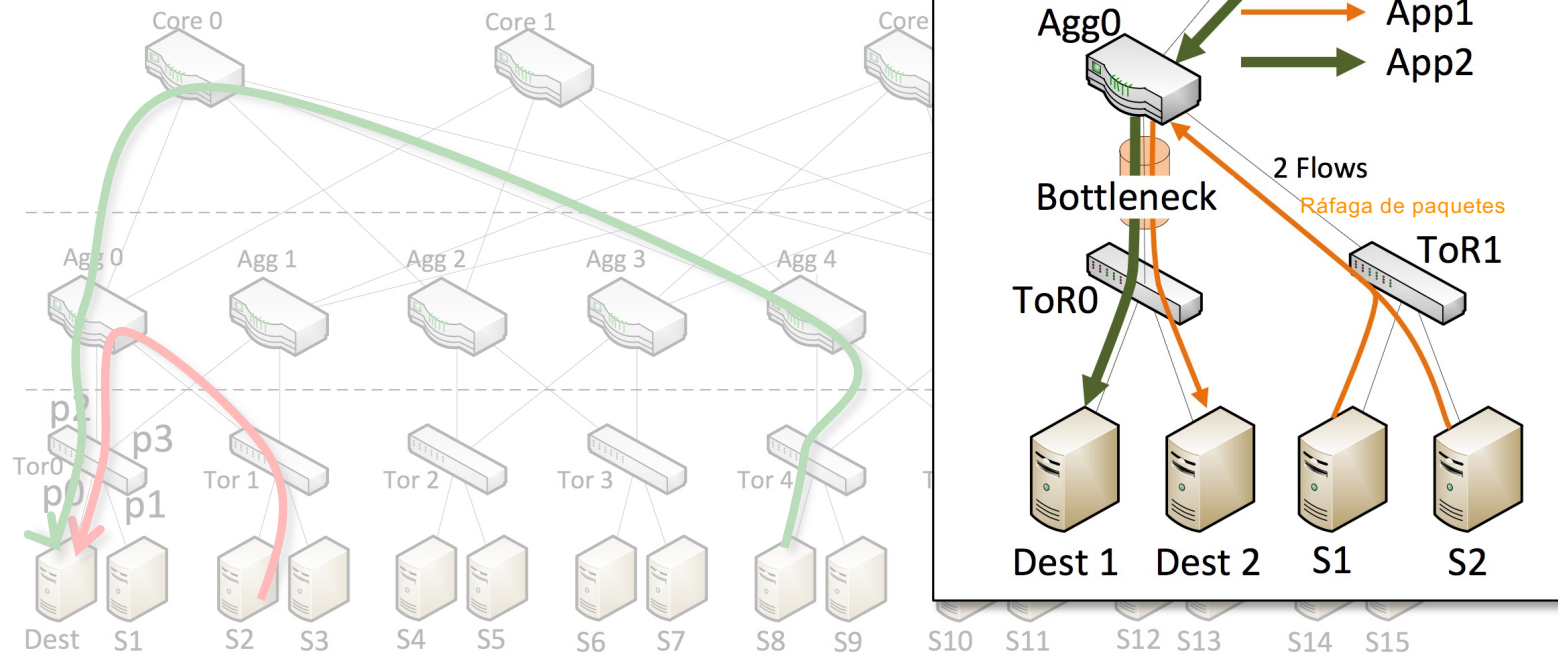
TCP Outcast

- Partition+Aggregate, drop-tail
- Muchos más flujos de servidores alejados que de cercanos, llegan por puertos diferentes
- Congestión en el cuello de botella → buffer lleno → pérdidas
- Lo más probable es que se de una llegada en ráfaga y se pierda entera
- Esa ráfaga probablemente viene de uno de los puertos de entrada
- O es el mux de paquetes de los dos



TCP Outcast

- En enlace con muchos flujos la ráfaga de pérdidas se distribuye entre muchos flujos
 - Baja probabilidad de perder toda una ventana y acabar en RTO
- En enlace con pocos flujos se distribuye entre pocos flujos
 - Mayor probabilidad de acabar en un RTO
- El resultado es que flujos con menor RTT obtienen menor goodput (inverso de lo habitual)



upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Outcast

Tráfico de datos en el DC

upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Redes de Nueva Generación
Área de Ingeniería Telemática

Tráfico de datos