# Arquitectura de routers

## *¿Cómo funcionan los routers?*

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

Arquitectura de Redes, Sistemas y Servicios
3º Ingeniería de Telecomunicación

# Temario

- Introducción
- Arquitecturas, protocolos y estándares
- Conmutación de paquetes
- Conmutación de circuitos
- Tecnologías
- Control de acceso al medio en redes de área local
- Servicios de Internet

# Temario

- Introducción
- Arquitecturas, protocolos y estándares
- Conmutación de paquetes
  - Principios
  - Problemas básicos
    - **Como funcionan los routers (Nivel de red)**
    - Encaminamiento (Nivel de red)
    - Transporte fiable (Nivel de transporte en TCP/IP)
    - Control de flujo (Nivel de transporte en TCP/IP)
    - Control de congestión (Nivel de transoporte en TCP/IP)
- Conmutación de circuitos
- Tecnologías
- Control de acceso al medio en redes de área local
- Servicios de Internet

# Problemas en el nivel de red

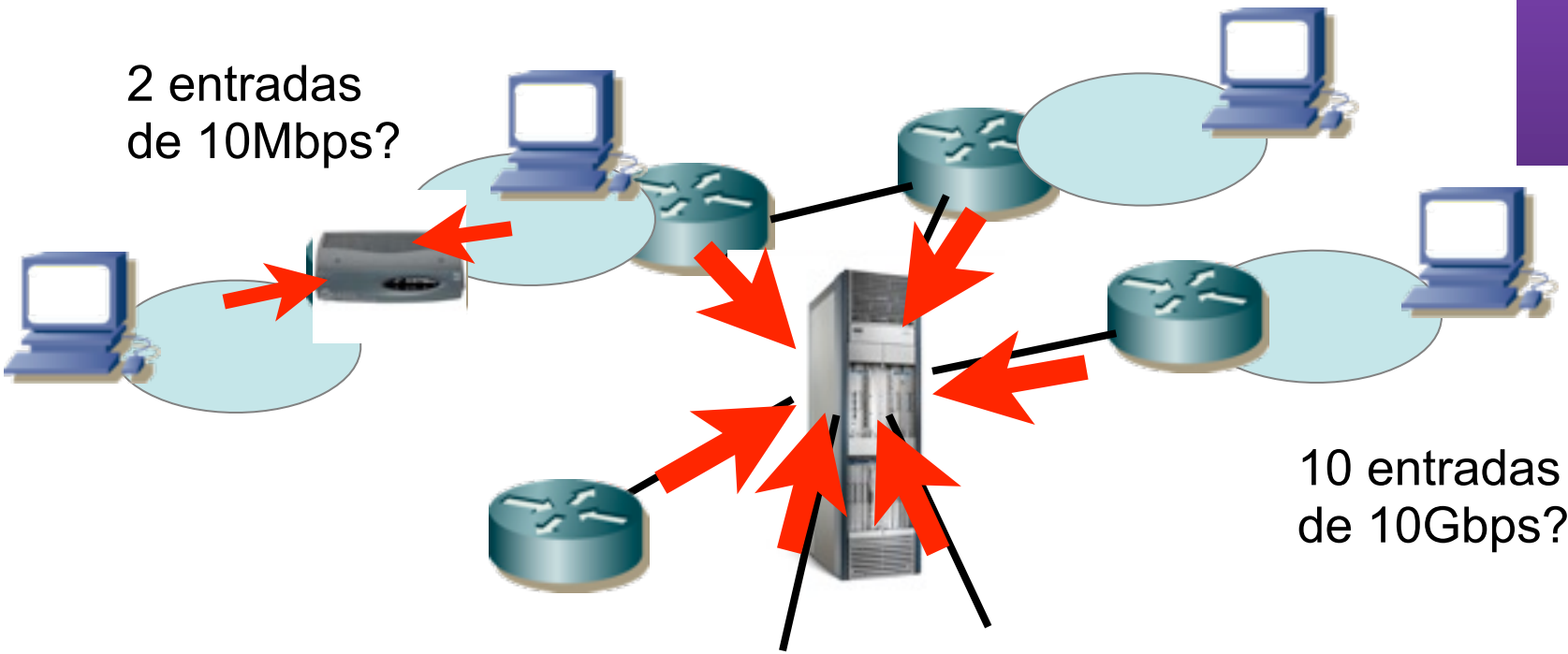
- Problema de enrutamiento:

  Cómo sé a qué vecino debo reenviar? Como llegan los paquetes hasta el destino?

- **Construcción de routers**

  Se pueden construir conmutadores/encaminadores de paquetes?

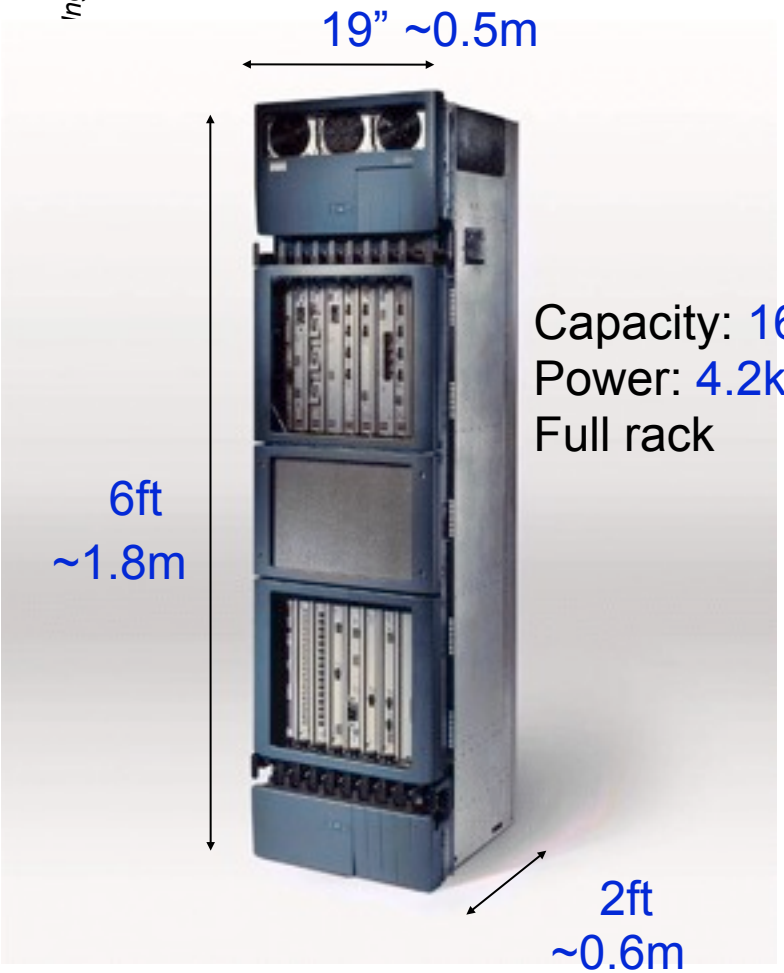
  Cómo de rápidos/eficientes? cuántos paquetes por segundo puedo reenviar? Qué funcionalidades necesitan?



# ¿Qué pinta tiene un router?

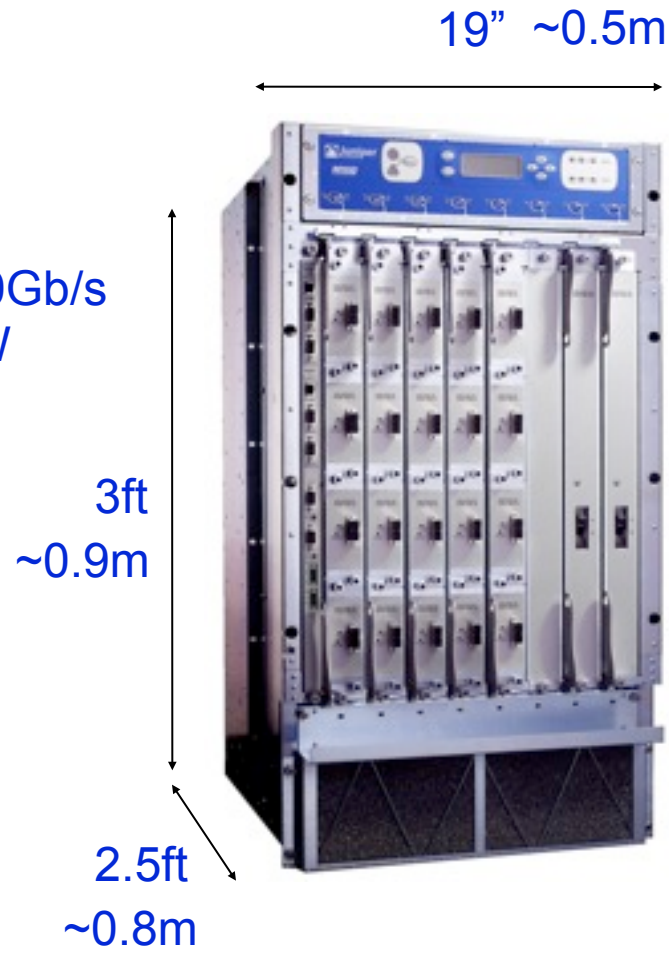
Ingeniería Telemática

## Cisco GSR 12416




Capacity: 160Gb/s
Power: 4.2kW
Full rack

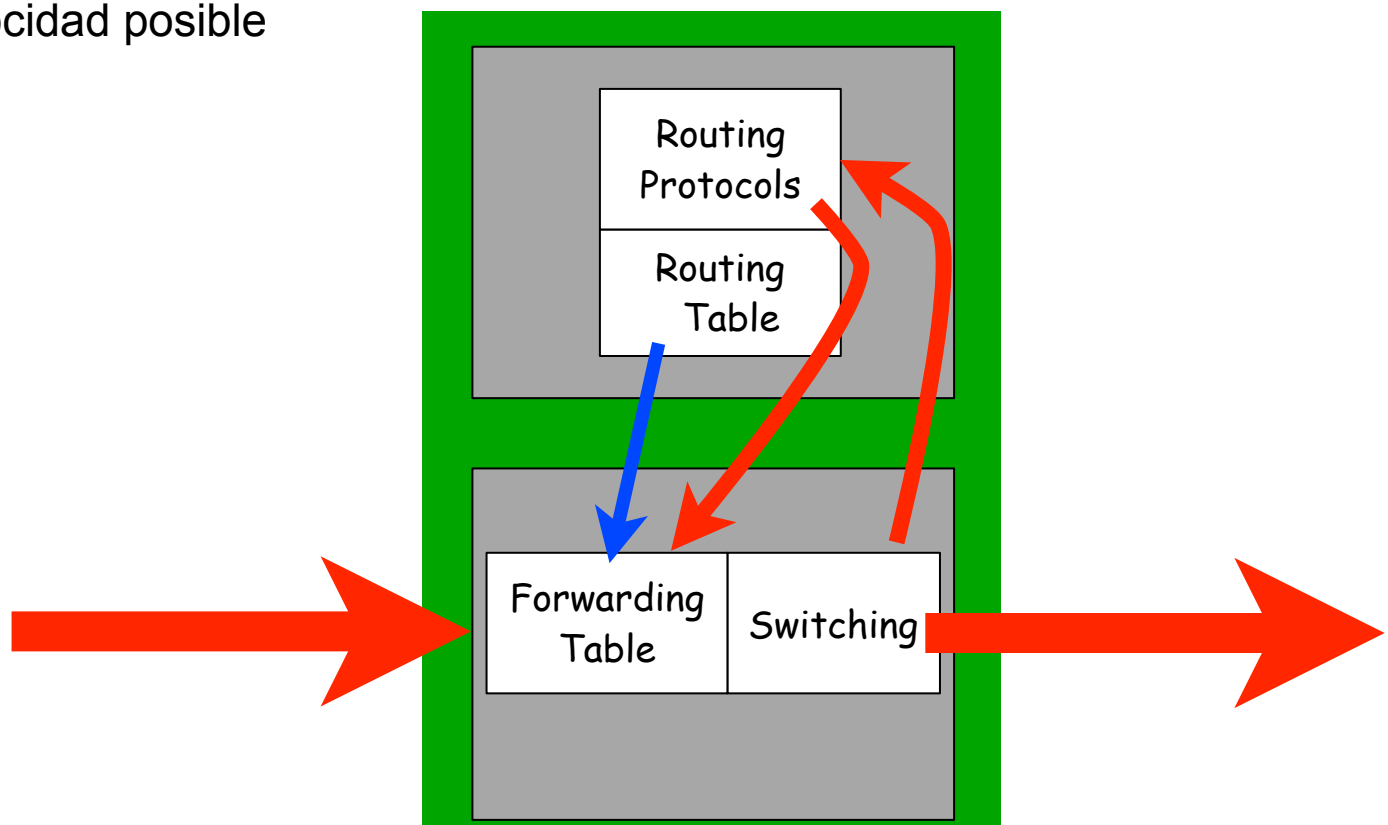
## Juniper M160




Capacity: 80Gb/s
Power: 2.6kW
Half-a-rack

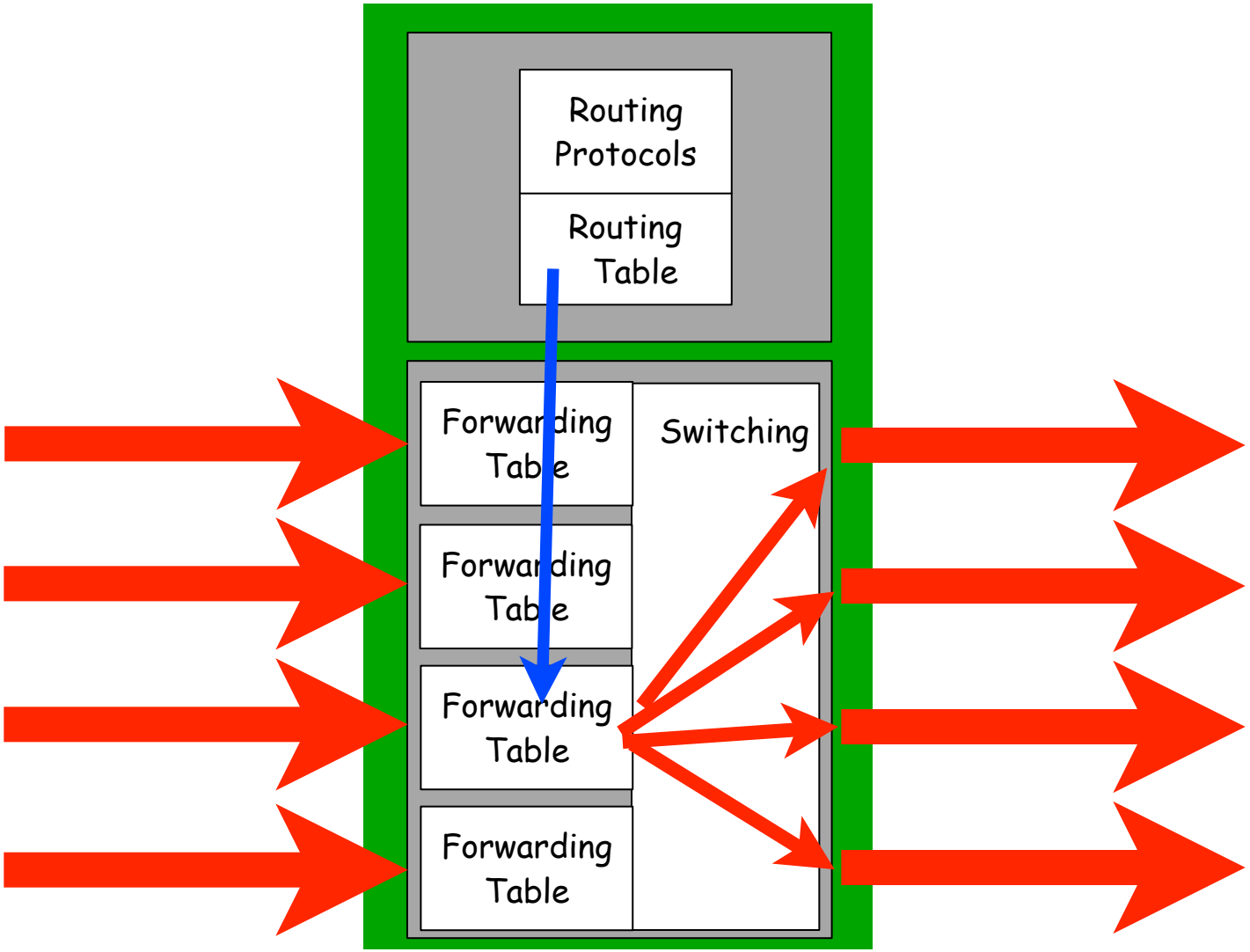
# Arquitectura básica de un router IP

- **Plano de control**
  - Los protocolos de enrutamiento y otra información de control pueden enviar y recibir información de la red
  - Construyen la tabla de rutas y configuran el plano de datos
- **Plano de datos**
  - Los paquetes que pasan por el router deben atravesarlo a la máxima velocidad posible



# Arquitectura básica de un router IP

- **Plano de datos**
  - No es una entrada y una salida sino varias entradas y salidas



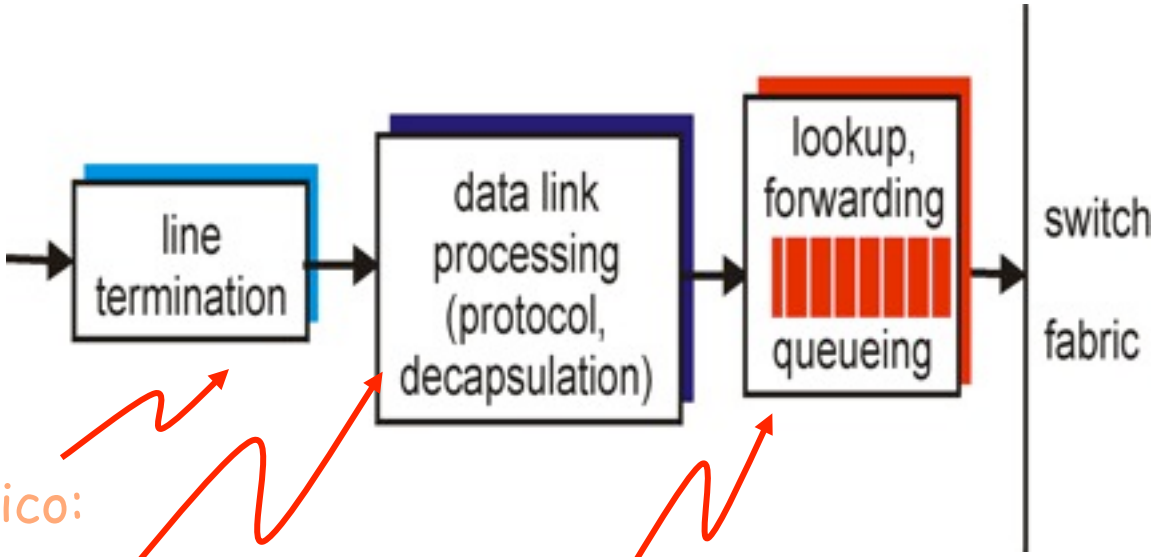
# Proceso por paquete

¿Qué hay que hacer por cada paquete?

- **Recibir** a nivel de enlace.
- **Lookup**: buscar la dirección destino del paquete en la tabla de reenvío (forwarding) para identificar por donde debe salir
- **Header processing**: manipular cabecera IP: decrementar TTL, recalcular checksum
- **Switching**: llevar el paquete a la tarjeta de salida correspondiente
- **Buffering**: almacenar el paquete durante los tiempos que deba esperar
- **Transmitir** a nivel de enlace

- La velocidad a la que se pueda hacer da las prestaciones

# Puertos de entrada

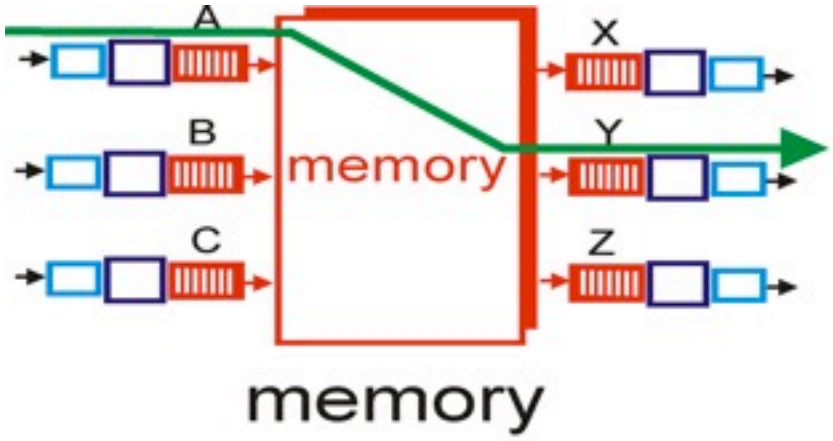



Nivel físico:
Recepción de bits
Nivel de enlace:
por ejemplo
Ethernet

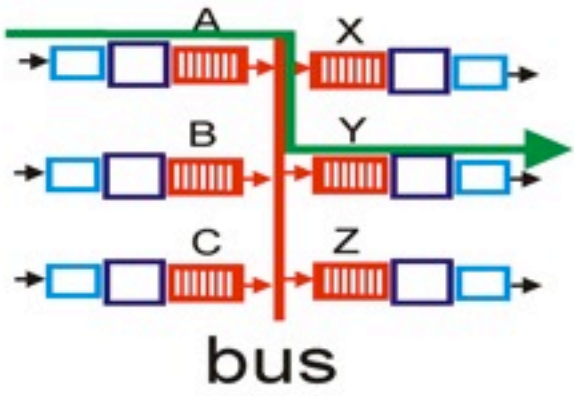
**Decentralized switching**:

- A partir de la dirección de destino buscar el puerto de salida en la tabla de reenvios (se mantiene una copia de la tabla de rutas en la memoria del puerto de entrada)
- objetivo: porcesar paquetes entrantes a velocidad de linea
- Cola de entrada: si los paquetes llegan a mas velocidad que la velocidad de entrada a matriz de conmutación

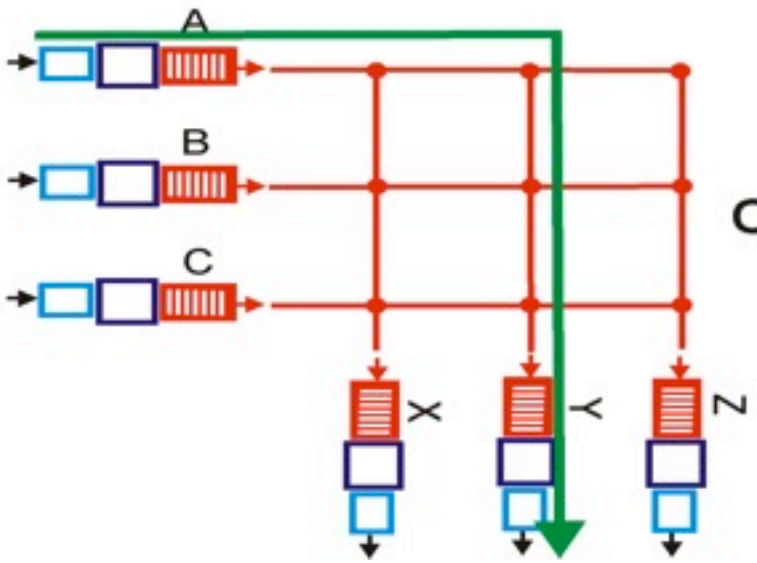
# Tipos de conmutación




memory

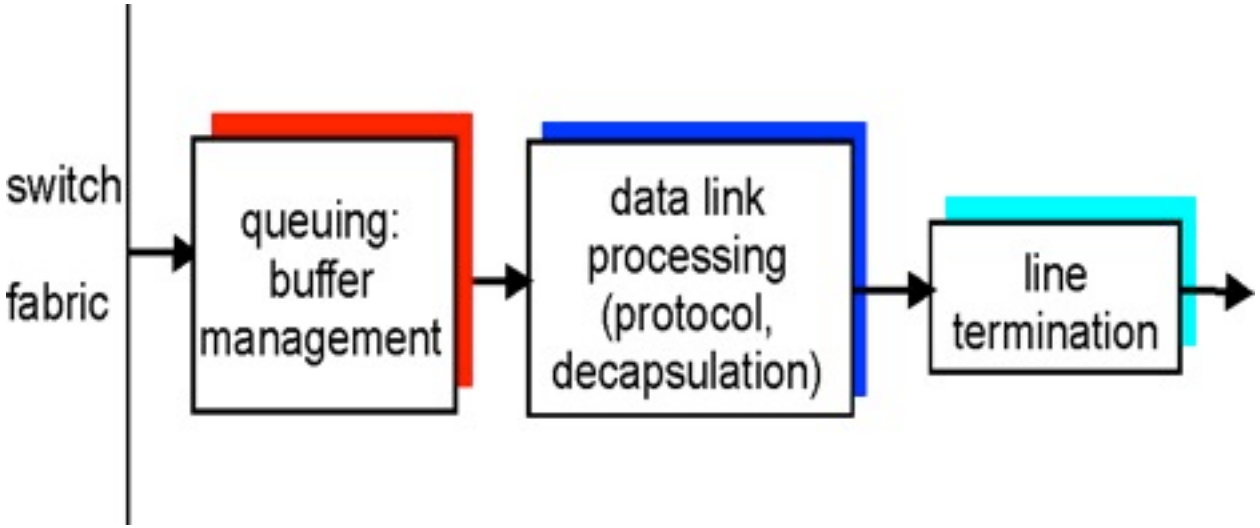



bus




crossbar

# Puertos de salida




- *Buffer (cola de salida)*necesario si los paquetes pueden llegar a mas velocidad que la del puerto de salida desde la matriz de conmutación
- *Planificación (Scheduling)* elige entre los paquetes disponible para transmision (FIFO, u otras?)

# Las dificultades

- En media debemos hacer todo el proceso para cada paquete en el tiempo en el que la tarjeta de red recibe el siguiente

  De lo contrario estamos acumulando paquetes

- Tiempo para enviar un paquete de 29+14bytes

  (ping de 1 byte) a 10Gbps : **34ns**

# Address lookup

- Buscar una dirección IP en una tabla de rutas

  Longest prefix match no vale con encontrar uno que cumpla

  La tabla puede tener ~150000 entradas

- Se usan estructuras prefix tree

  Ejemplo: almacena los prefijos

  {*,000*,0000*,0001*,100*,110*,111*}

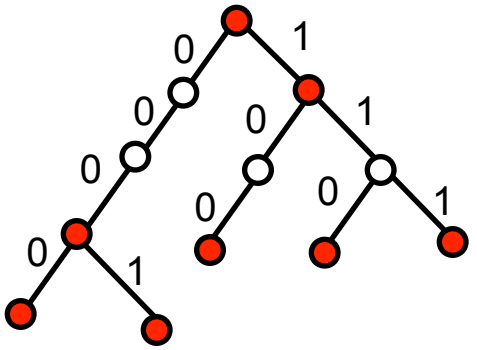
  busquedas de tipo

  00010010 -> 0001*

  10101001 -> 1*

  00101010 -> *

  11010101 -> 110*




Se hace en hardware (NSE network search engines)

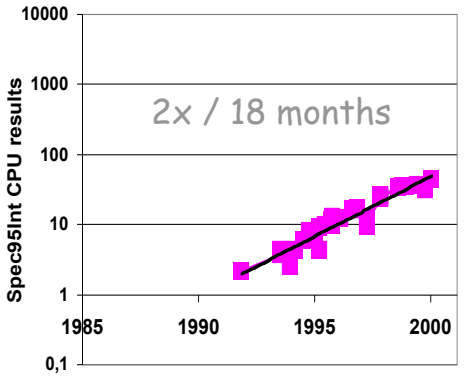
(Se pueden encontrar circuitos que hacen 100000000 busquedas por segundo ~10ns)

# Almacenamiento y cálculo

- No es trivial almacenar y recuperar paquetes en 30ns velocidades de acceso a memoria?
- No es trivial manipular la cabecera

  Hardware especifico para routers de alta velocidad
- Problemas con la ley de Moore
  - Velocidad de procesado se duplica cada 18 meses
  - Capacidad de la fibra se duplica cada año
  - Velocidad de la memoria solo se multiplica por 1.1 cada 18 meses

Packet Processing Power




- El tráfico de Internet crece más deprisa

  **Capacidad típica de los routers comerciales**

  Capacidad 1992 ~ 2Gb/s

  Capacidad 1995 ~ 10Gb/s

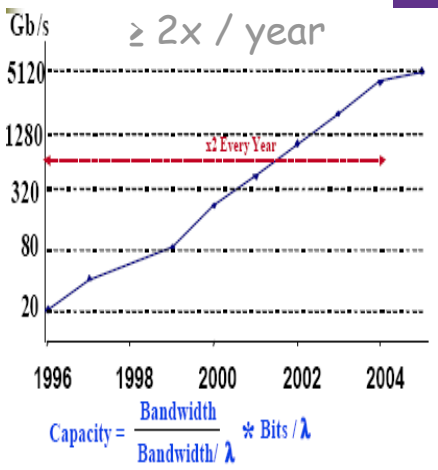
  Capacidad 1998 ~ 40Gb/s

  Capacidad 2001 ~ 160Gb/s

  Capacidad 2003 ~ 640Gb/s

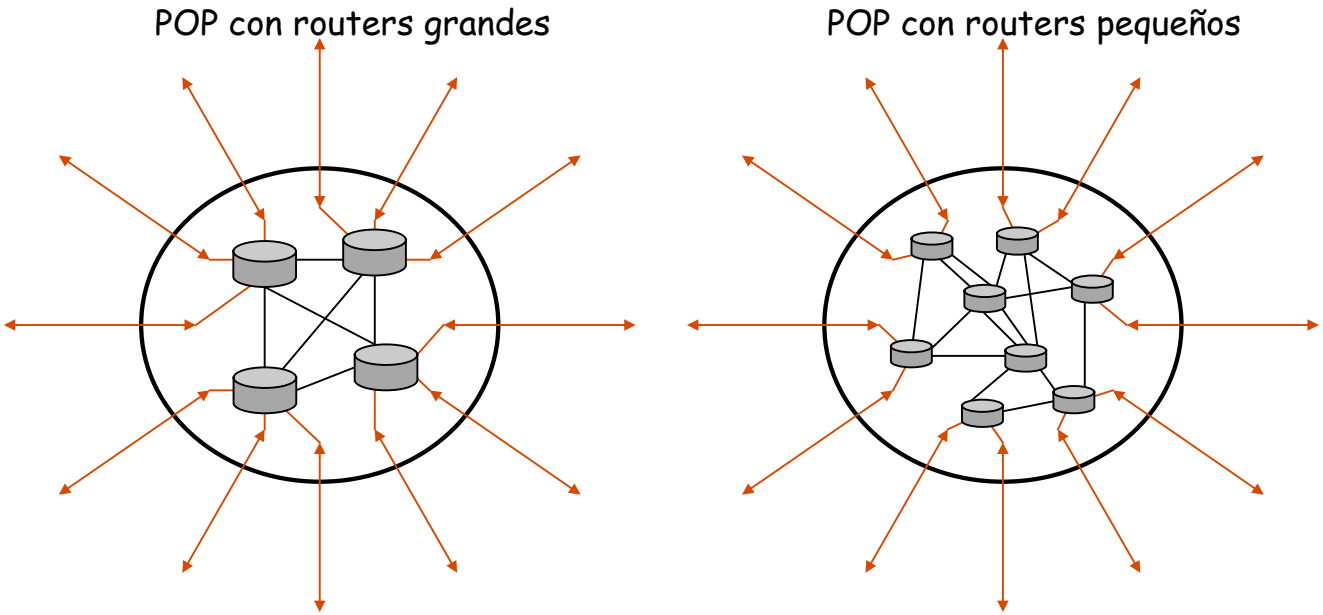
  Media approx 2.2x / 18meses

Single Fiber Capacity (commercial)



# Distribuyendo la carga

- Siempre podemos usar mas routers e interconexiones para distribuir el tráfico
- Interesa más tener pocos routers de gran potencia
  - Gestión más simple
  - Menos consumo y menos coste: lo que cuesta son los puertos




- Puertos: precio >$50k,  consumo > 400W.
- Alrededor de  50-60% de los puertos es para interconexión

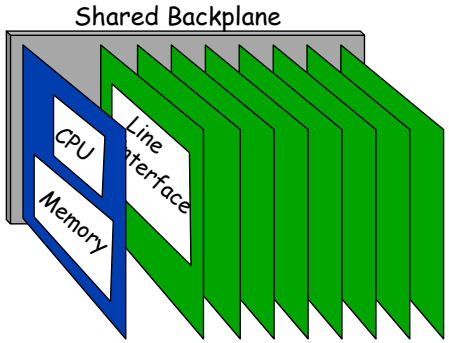
# Switching

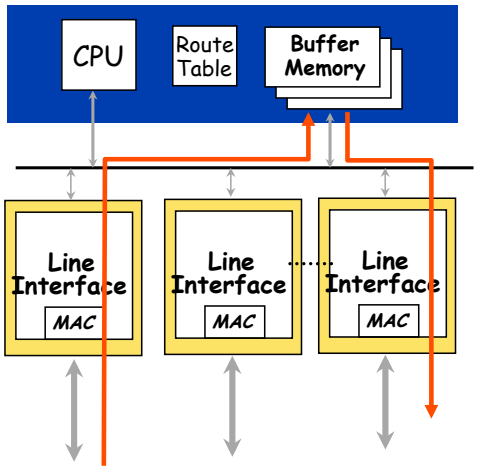
- Problema fundamental
- ¿Como enviamos los paquetes de la entrada a la salida correspondiente?

  Matriz de conmutaciñon / switch fabric
- El problema aparece en conmutación de circuitos primero
  - En telefonía es más fácil el circuito se establece con antelación. Una vez establecido todos los datos que llegan por un puerto de entrada van al mismo puerto de salida hasta que se libera
- Necesitamos el mismo tipo de sistemas pero que sean capaces de cambiar de estado en el tiempo de un paquete
  - La velocidad a la que los paquetes atraviesen esta matriz de conmutación es fundamental

# Switching architectures

- Memoria central y bus
  - Ordenador tradicional conmutación bajo control de la CPU
  - El paquete se copia del puerto de entrada  a la memoria
  - El paquete se copia de la memoria al puerto de salida
  - Gran flexibilidad y fácil de programar
  - El paquete pasa 2 veces por el bus
  - La máxima velocidad a la que reenviamos paquetes es la mitad de la velocidad del bus
  - Los paquetes deben esperar en las tarjetas de entrada (Contención)




  - Routers de primera generación

    Tipica velocidad agregada < 0.5Gbps

# Switching architectures

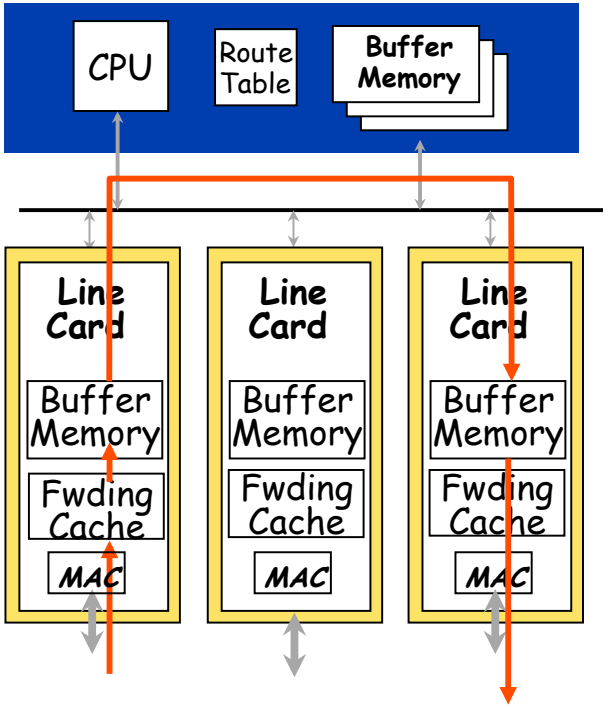
- Bus compartido y puertos independientes
  - El paquete se copia de la memoria del puerto de entrada a la del puerto de salida
  - Más complejo
  - El paquete pasa 1 vez por el bus. La máxima velocidad agregada es la del bus
  - Routers de segunda generación

    Tipica velocidad agregada < 5Gbps
  - Pero en un instante como mucho

    pasa 1 paquete por el bus

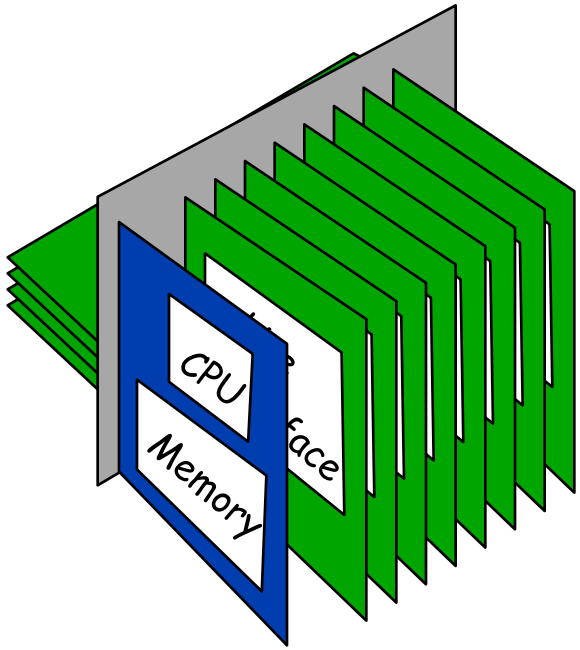
    si otro puerto de entrada tiene un paquete

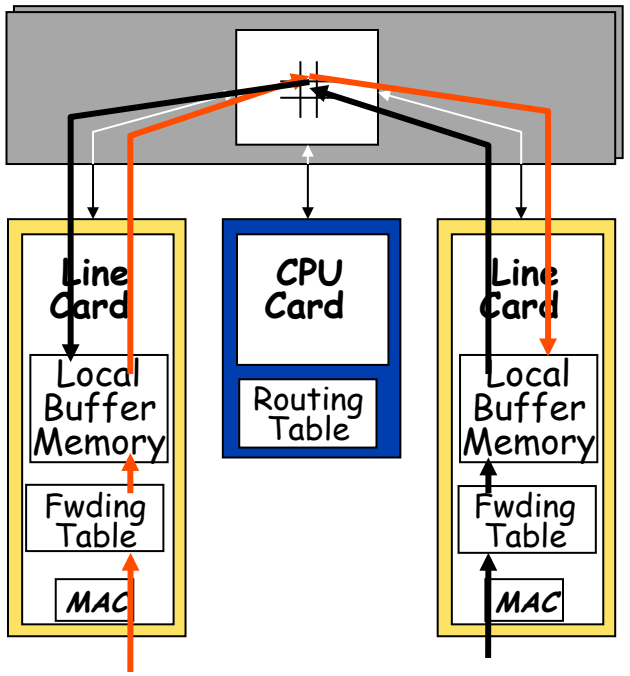
    debe esperar...



# Switching architectures

- Tercera generacion ++
  - Matriz de conmutación más compleja capaz de tener varias transferencias a la vez (switched backplane)
  -

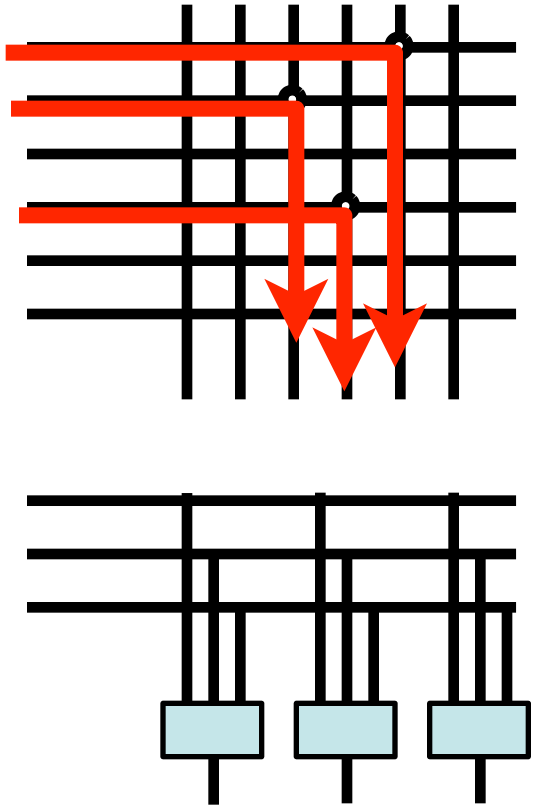


# Switching

- Switch fabrics capaces de intercambiar varios flujos al mismo tiempo
- **Crossbar (space-switching)**
  - N veces mas rapido que un bus
  - Pero hay problemas si dos quieren enviar un paquete a la vez al mismo destino (contencion)

    O bien el acceso al puerto de salida es N veces más rápido o bien puede haber bloqueo
- **Broadcast**
  - N veces mas rapido que un bus
  - El acceso al puerto de salida N veces mas rápido o bloqueo
- Necesita muchos recursos
- El cuello de botella es la entrada al puerto de salida
- Escalan mal con el numero de puertos



# Switching

- **Redes Banyan**

  Facil de encaminar paquetes por la electronica

  Electronica a la velocidad del puerto de entrada

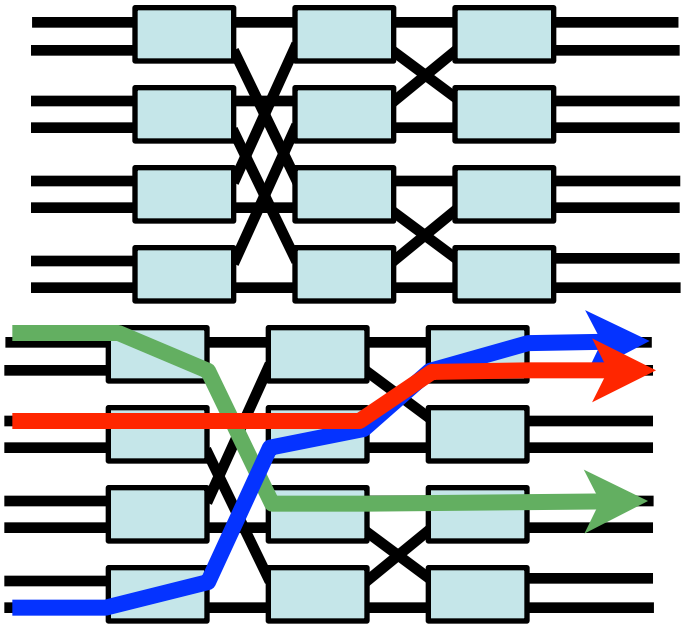
  Pero por si sola tiene probabilidad de bloqueo interno (bloqueo incluso aunque no quieran salir por el mismo puerto)

- El bloqueo interno se puede eliminar si ordeno las entradas

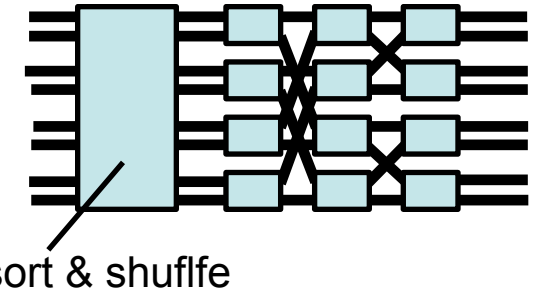
  Batcher-Banyan networks

- Pero si dos paquetes quieren ir a la vez a la misma salida solo hay dos posibilidades
  - Retardarlo con buffers
  - Hacer el puerto de salida mas rápido

Banyan network de 3 bits



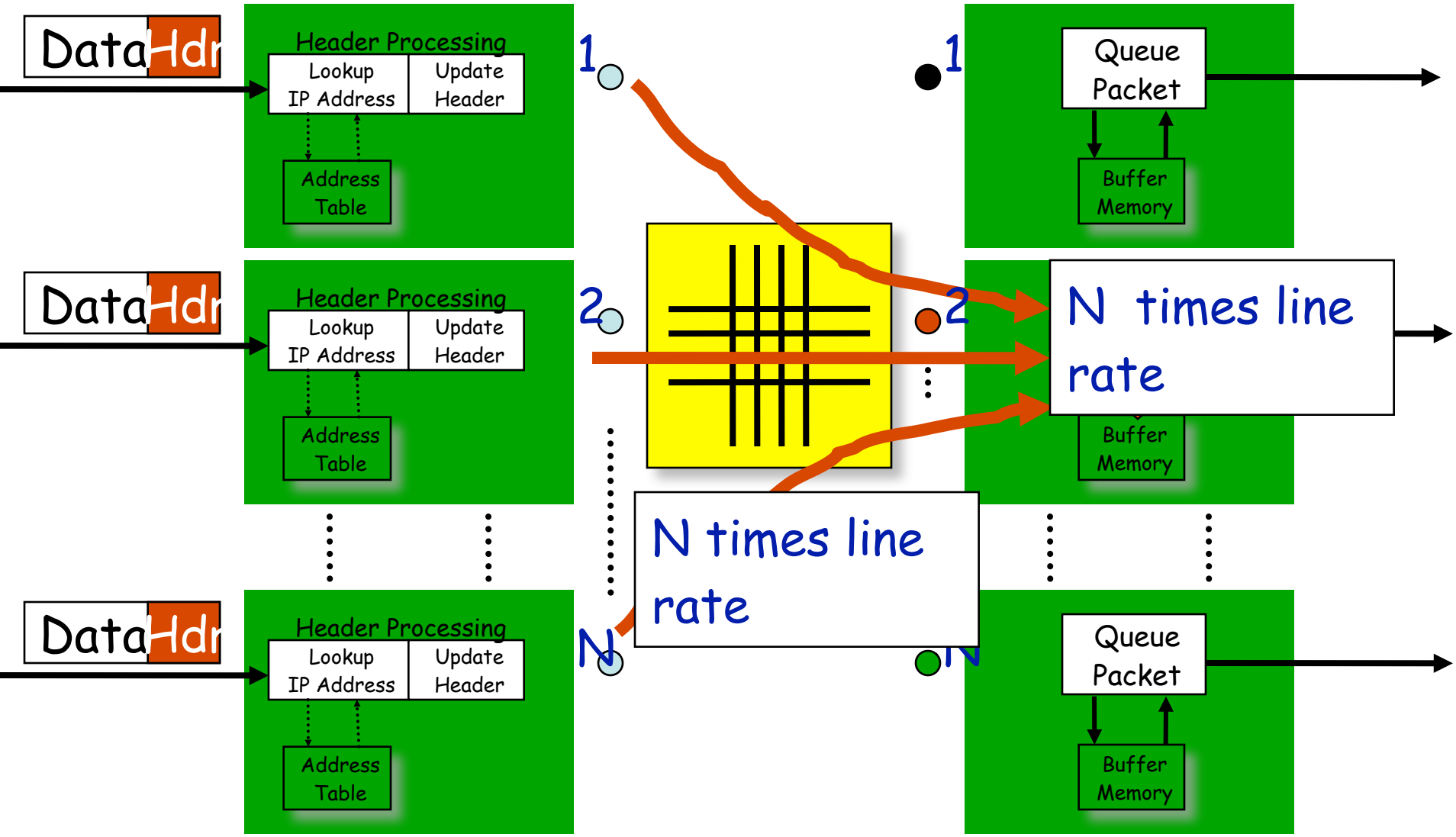
Batcher-Banyan



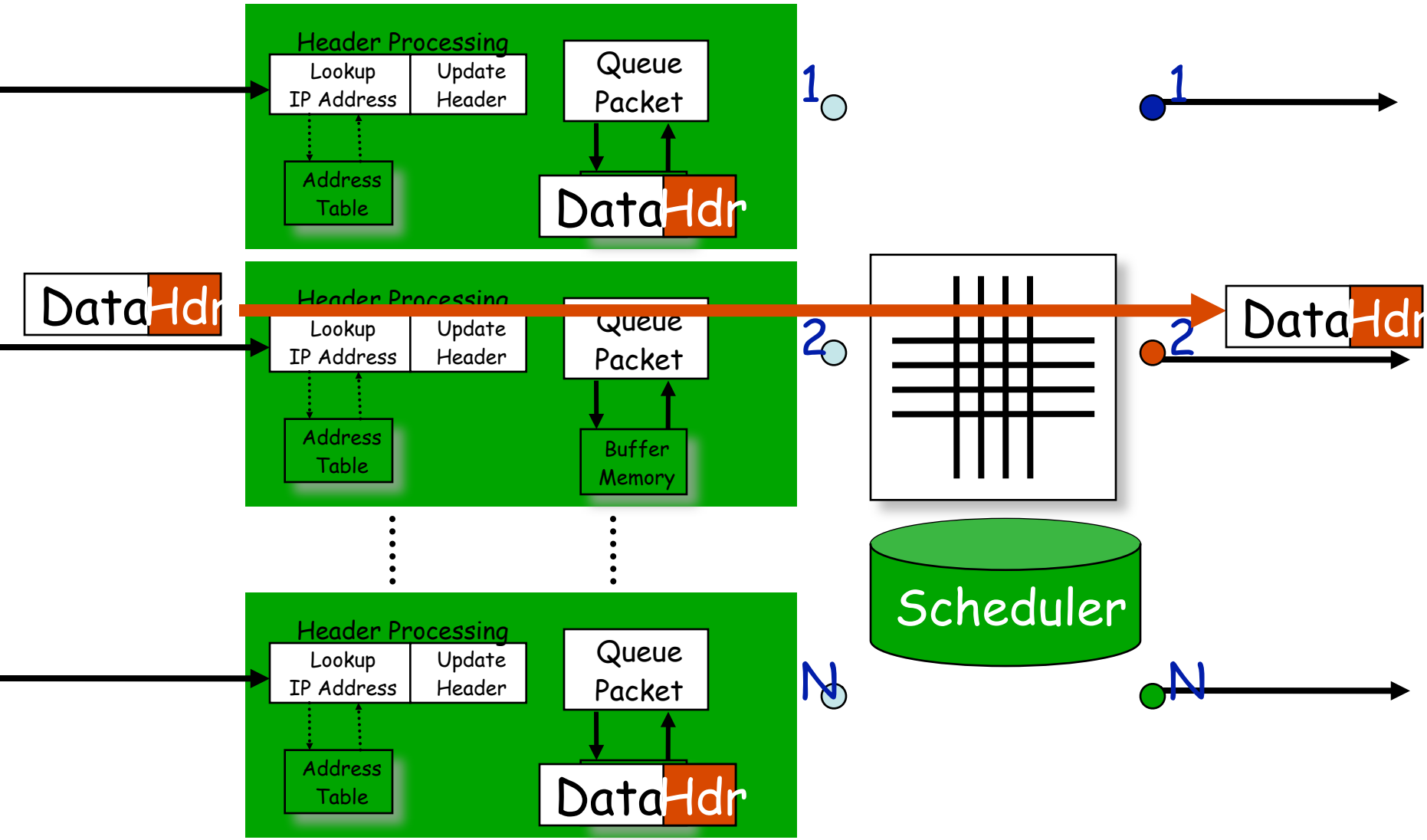
# Arquitecturas de routers

- Dos filosofias
  - Colas a la entrada
  - Colas a la salida
-

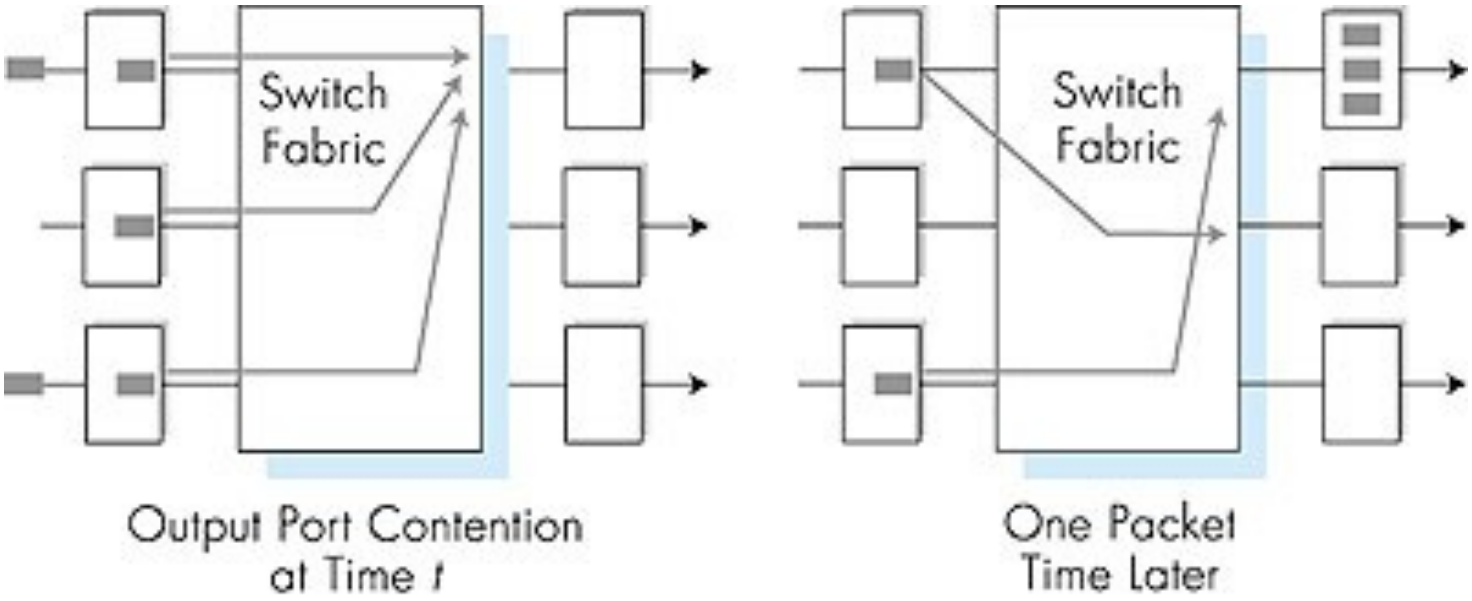
# Colas a la salida



# Colas a la entrada



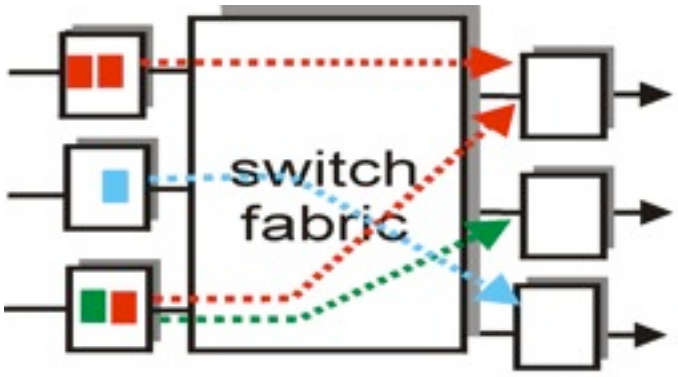
# Usando colas en los puertos de salida



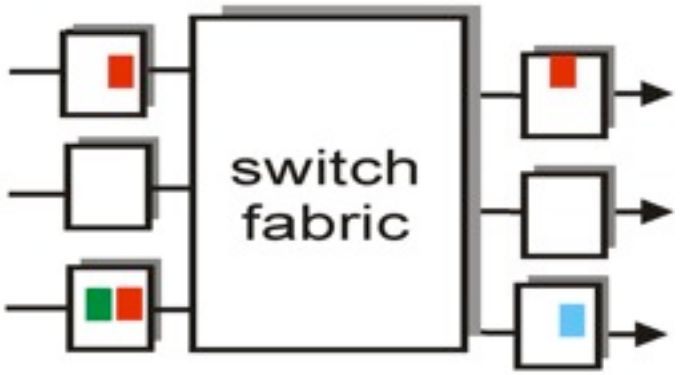

- Cuando la velocidad de llegada a traves de la matriz de conmutación es superior a la velocidad de salida se almacenan en buffer (cola de salida)
- *almacenamiento (retraso) y pérdidas debidas al desbordamiento del buffer del puerto de salida*
- Problema: *matriz de conmutación debe ser N veces mas rápida que el puerto de entrada/salida*

# Usando colas en los puertos de entrada

- Matriz de conmutación más lenta que la suma de los puertos de entrada -> hay que almacenar en el puerto de entrada
- Bloqueo Head-of-the-Line (HOL): un paquete que no puede ir a un puerto bloquea al resto de paquetes en el puerto de entrada, aunque podrían ser servidos
- *Retraso y pérdidas debidas a desbordamiento de puerto de entrada*
- Combinación de las dos técnicas para conseguir poco HOL con matrices de conmutación rápidas pero no tanto como la suma de los puertos




output port contention
at time t - only one red
packet can be transferred




green packet
experiences HOL blocking

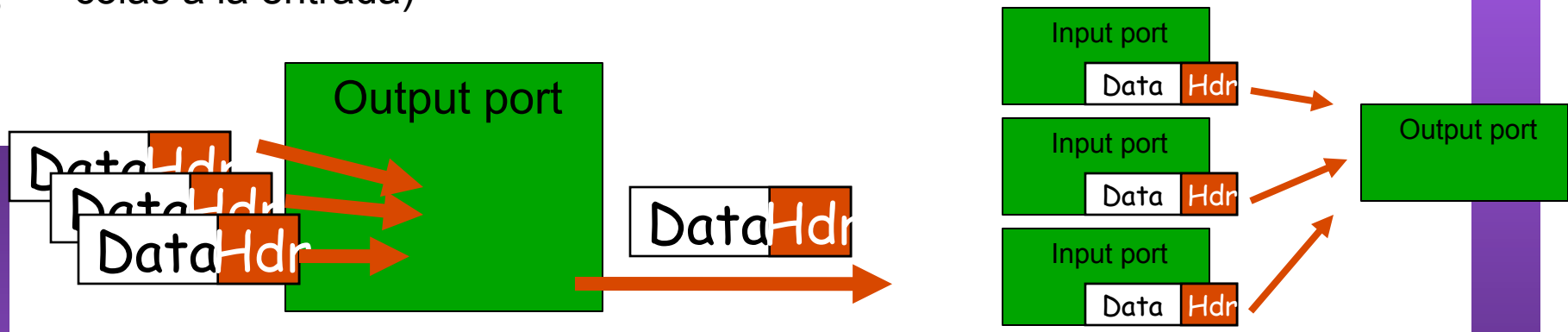
# Arquitecturas de routers

- Colas a la salida
  - Al puerto de salida pueden llegar durante un tiempo mas paquetes de los que puede servir
  - Se plantea el problema de la elección de que paquete envío cuando se me acumulan
  - El problema se llama planificación (scheduling)
- Colas a la entrada
  - Al puerto de salida llegan paquetes a la velocidad de linea por lo que no tiene mucho donde elegir...
  - La planificación es necesaria para decidir cual de los puertos de entrada debo dejar que pase a la salida

- **Planificación/scheduling es un problema fundamental de redes**
  - Como elijo que cliente/usuario/entrada/flujo debo servir
  - Como gestiono la cola de peticiones

# Planificación (scheduling)

- Tráfico variable. Paquetes llegan al puerto de salida más rápido de lo que pueden ser enviados (o hay paquetes en varios puertos de entrada con colas a la entrada)




- No podemos mantener esta situación por tiempo ilimitado
- Pero podemos amortiguar ráfagas de cierta duración (colas)
- Tenemos varios candidatos de paquetes a enviar para un recurso limitado (sólo podemos enviar uno)

  ¿Cómo decidimos cual se envía y cual espera?

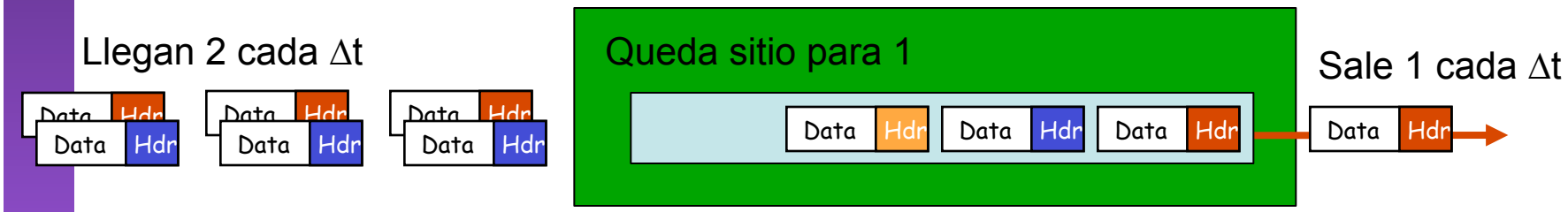
  ¿Qué hacemos con los paquetes que llegan cuando el almacenamiento es escaso?

  Problema de **planificación** (o **scheduling**)

# Planificación (scheduling)

- Aproximación más simple: por orden de llegada (FCFS First came first served)

  Si un paquete llega y no hay sitio se elimina (DropTail)
- Problemas tipicos
  - El orden de llegada puede ser arbitrario (llegar por el primer puerto de entrada?)
  - **Problemas de injusticia entre flujos**




- **A algunas aplicaciones les importa más el retardo que a otras**

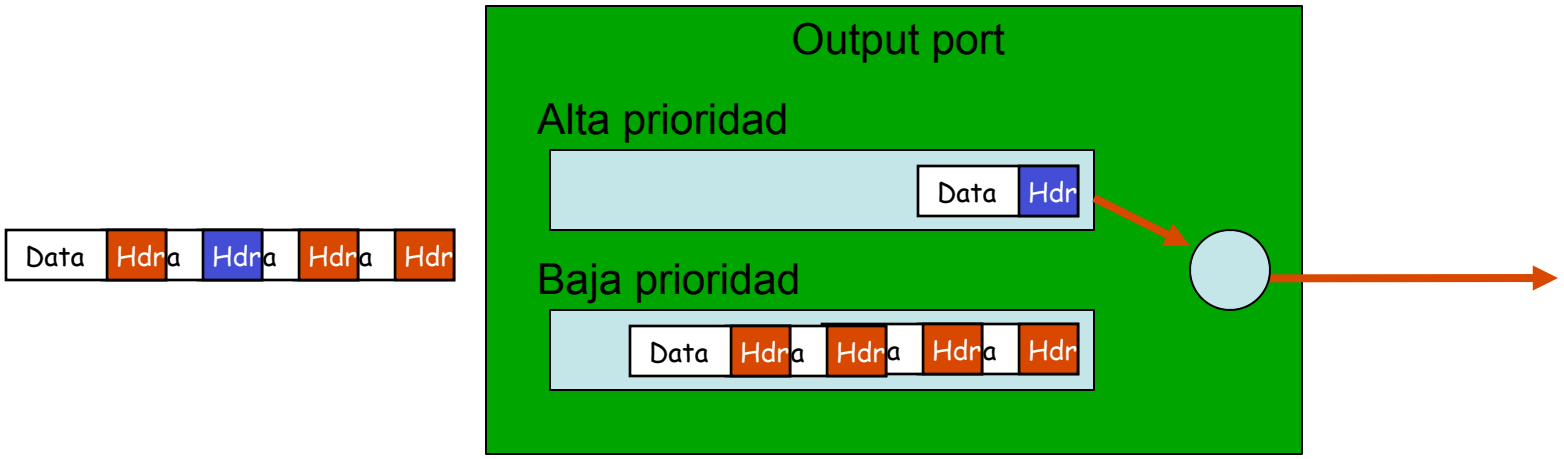


Voz sobre IP pocos paquetes pero muy importante que lleguen rápido (>150ms inutil)


Descarga de fichero muchos paquetes pero no importa que se retrasen muchos milisegundos

# Planificadores

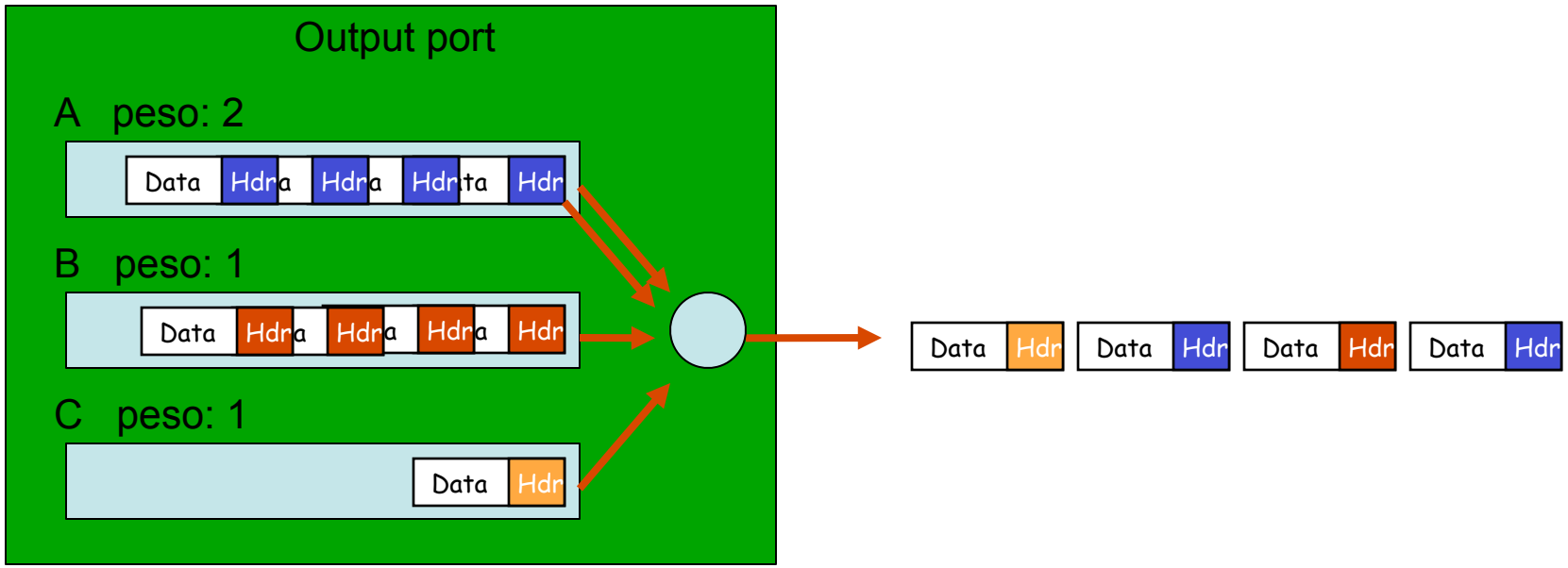
- Colas con prioridad (PQ Priority Queueing) y clases de aplicaciones
- Solo envía de las colas con menos prioridad si no hay nada en las de mas prioridad (con 2 o más niveles)




- Normalmente aplicaciones con requisitos de retardo vs aplicaciones normales. Y algún otro nivel más pero no muchos
  - Limitando clasificación del tráfico
  - Limitando estado interno (mayor complejidad con más colas)

# Planificadores

- Planificadores justos, reparten el envío por igual entre varias clases de paquetes: GPS (Generalized Processor Sharing), WFQ (Weigthed Fair Queueing)
- Incluyendo reparto a partes proporcionales (por ejemplo: que la clase A tenga el doble de procesador que la B)



# Conclusiones

- Arquitectura de routers con plano de control y plano de datos
- Problemas que deben resolverse para construir el plano de datos
  - Address lookup (prefix-trees)
  - Acceso a memoria y calculo para manipulación de cabeceras
  - Conmutación de paquetes en el plano de datos
    Switching fabrics: buses, crossbar, broadcast, banyan...
  - Colas a la entrada o a la salida
  - El problema del scheduling

- Proxima clase: **encaminamiento / routing**
  ¿como hacemos para saber como se llega a cada destino?