CONTROL DEL FLUJO DE INFORMACIÓN

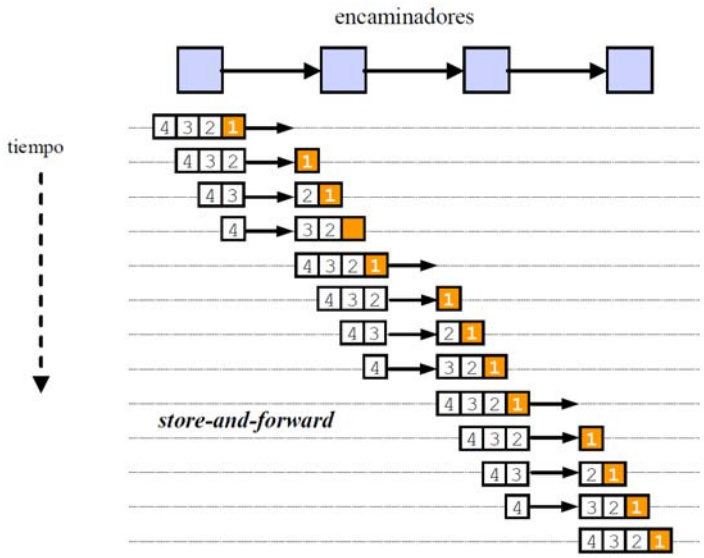
Los paquetes avanzan en la red de encaminador a encaminador hasta llegar a su destino. Pero, ¿cómo se efectúa el avance? ¿Qué hay que hacer si los recursos que necesita un paquete para seguir adelante están ocupados? Este tipo de cuestiones se conocen como "control del flujo" de los paquetes.

1.- Avance de los paquetes: SF, WH, CT

¿Cómo se transmite un paquete de encaminador a encaminador? Supongamos que los enlaces de la red son de un byte, con lo que en un "ciclo" de transmisión se pasará un byte entre dos encaminadores. Los paquetes que hay que transmitir van a ser bastante más largos, por lo que necesitaremos varios ciclos de transmisión para pasar todo el paquete (todos los *flits* o bytes) de un encaminador al siguiente. Mientras tanto, ¿qué hay que hacer con un paquete que se está recibiendo? Veamos las dos alternativas principales: *store-and-forward* y *wormhole / cut-through*.

1.1.- Store-and-forward (SF)

Store-and-forward es el nombre de la técnica empleada en la primera generación de multicomputadores para pasar los paquetes entre dos encaminadores sucesivos (técnica habitual en las redes LAN y WAN).



En este caso, el encaminador que está recibiendo el paquete no hace nada con el mismo hasta que no lo termina de recibir del todo. A continuación, analiza la cabecera del paquete y decide por dónde debe retransmitirlo. De esa manera, la transmisión de los paquetes se limita siempre a dos encaminadores: el que transmite y el que recibe. Mientras se produce la transmisión, el paquete se guarda en un búfer interno del encaminador (véase la figura siguiente).

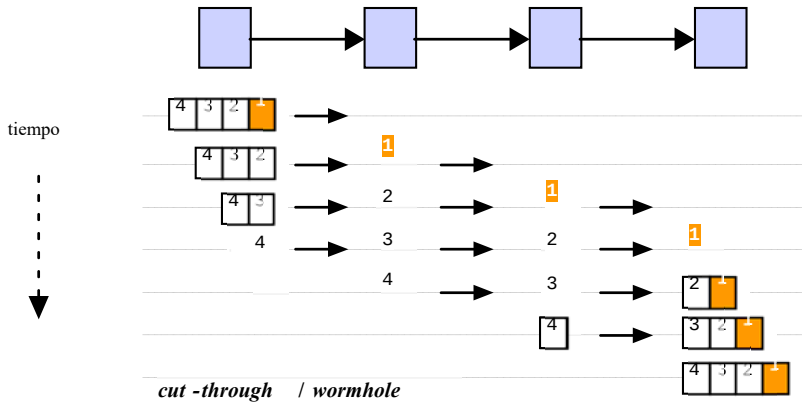
En primera aproximación, el tiempo de comunicación de un paquete resulta proporcional al tamaño del paquete (L) y a la longitud del camino recorrido (d):

$$T_{sf} \sim L \times d$$

Así pues, cuando se duplica la distancia a recorrer, también se duplica la latencia de un paquete.

1.2.- Wormhole (WH) / Cut-through (CT)

La técnica SF no es muy adecuada para los sistemas MPP, ya que la latencia de la comunicación puede resultar muy elevada (en función de la distancia a recorrer), y la comunicación es crucial en muchas de las aplicaciones que se ejecutan en paralelo. Por ello, los sistemas paralelos actuales utilizan otra técnica conocida como *wormhole* (WH) o *cut-through* (CT).



Cuando un encaminador recibe el primer *flit* de la cabecera de un paquete (normalmente una parte del registro de encaminamiento) analiza a dónde se dirige. Si debe continuar con su recorrido, entonces se le asigna directamente un puerto de salida y se comienza a retransmitir al siguiente encaminador, sin esperar a recibir todo el paquete; el resto de *flits* del paquete se enviará tras el primero, por el mismo camino, según vayan llegando. Así, el paquete queda distribuido a lo largo del camino hacia el destino: la comunicación se ha "**segmentado**", ya que muchos encaminadores toman parte simultáneamente en la transmisión de un paquete.

Se necesitan d ciclos para que el primer *flit* de la cabecera del paquete llegue al destino, tras lo cual llegarán, ciclo a ciclo, el resto de *flits* del paquete. Por tanto, en primera aproximación, la latencia de un paquete en modo WH o CT será proporcional a la suma de L y d. Es decir,

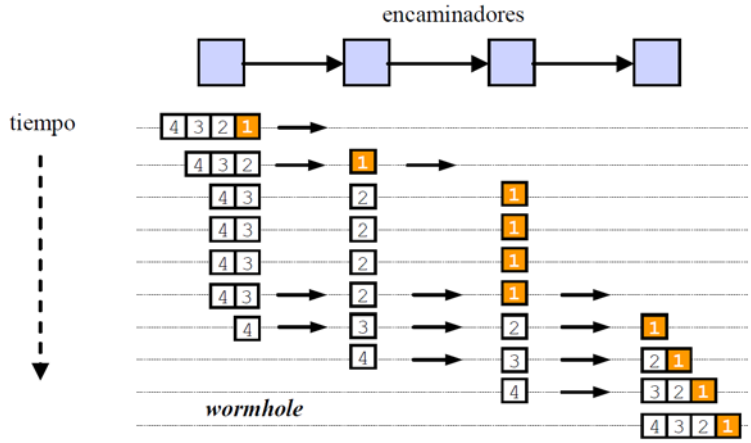
$$T_{ct/wh} \sim L + d$$

Claramente, la latencia de la comunicación en modo *wormhole* será menor que en modo *store-and-forward*, ya que L y d se suman en lugar de multiplicarse: el efecto de la distancia a recorrer en el tiempo de transmisión es mucho menor en modo CT/WH que en modo SF.

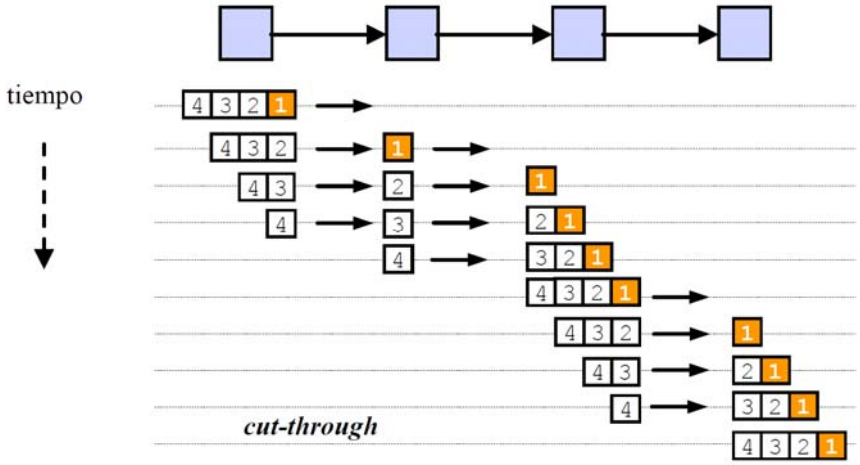
La transmisión de un paquete se efectúa de la misma manera en modo *wormhole* que en modo *cut-through*, pero si la cabecera de un paquete no puede continuar avanzando, porque la salida que necesita utilizar está ocupada por otro paquete, la respuesta es diferente en ambos casos:

- ***Wormhole***. Se detiene la recepción del paquete, y éste queda bloqueado a lo largo de todo el camino que recorre. Cuando se libera el camino de salida, se retoma la transmisión de los *flits* en todos los encaminadores implicados.

  El paquete que se ha parado va a mantener ocupados muchos recursos de la red, por lo que es posible que aumenten los conflictos entre paquetes.



- ***Cut-through***. Aunque la cabecera del paquete no pueda avanzar, se siguen recibiendo el resto de los *flits* del paquete, y se almacenan en un búfer propio del encaminador hasta que el canal de salida que se necesita se libere, en cuyo caso se proseguirá con la transmisión de la cabecera del paquete. De esta manera, el paquete cuya cabecera no puede avanzar no mantiene ocupados tantos recursos de la red, con lo que se producirán muchos menos conflictos con otros paquetes.



Si se utiliza *cut-through*, es necesario que los encaminadores dispongan de búferes con capacidad para almacenar paquetes de manera transitoria, hasta que prosigan su camino; si utilizamos *wormhole*, ese espacio, en principio, no es necesario (bastaría con poder guardar un *flit*: el que se está procesando en ese momento). En muchas máquinas se aplican estrategias intermedias entre WH y CT: no hay capacidad para almacenar un paquete completo, pero pueden guardarse varios *flits* (por ejemplo, pueden almacenarse 4 *flits* de un paquete a la espera de que quede libre el puerto de salida).

Analizado desde otro punto de vista, podemos decir que CT representa un compromiso entre WH y SF. Cuando hay poco tráfico en la red, CT se comporta igual que WH: no habrá conflictos en el uso de los recursos y, por tanto, no se almacenan los paquetes en los encaminadores intermedios. En cambio, cuando el tráfico es elevado CT se acerca al comportamiento de SF: los conflictos son muy habituales en los encaminadores de la red, y los mensajes se almacenan casi siempre.

2.- Conflictos en el uso de recursos: los búferes

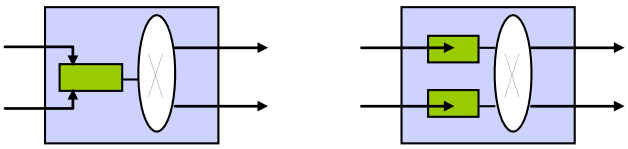
Los paquetes recorren la red utilizando los enlaces de la misma y los encaminadores de mensajes. La red es por tanto un recurso compartido, que va a ser utilizado simultáneamente por muchos paquetes. ¿Cómo hay que afrontar los conflictos que, inevitablemente, se van a producir al usar esos recursos? Ya hemos comentado en qué difieren CT y WH al tratar los conflictos: el paquete que no puede seguir adelante se almacena temporalmente en el encaminador intermedio, o se deja bloqueado a lo largo de toda la red.

En el caso de CT (y, por definición, de SF) se necesita de un "poco" de memoria para almacenar transitoriamente los paquetes (o algunos *flits* de un paquete) en los encaminadores, hasta que la salida correspondiente esté libre. Normalmente, los encaminadores no disponen de gran cantidad de memoria para almacenar mensajes, sino que tienen la posibilidad de almacenar unos cuantos bytes o unos cuantos paquetes. Por ejemplo, si los paquetes son de 64 bytes, sería suficiente con una capacidad de 1 o 2 kB (16 o 32 paquetes) por canal; no parece razonable tener una memoria de 1 MB para gestionar un posible bloqueo de 16000 paquetes. Además, el funcionamiento de los encaminadores debe ser eficiente y rápido, con el fin de que la latencia de los paquetes sea lo menor posible, y para ello lo más adecuado es que los encaminadores sean lo más sencillos posibles.

Hay muchas maneras de organizar y gestionar esos búferes (normalmente en forma de cola FIFO), y cada una de ellas tiene sus ventajas e inconvenientes. Además, sea cual sea el número de búferes, siempre es posible que una aplicación genere un tráfico muy intenso en un momento dado, que sature la capacidad de guardar más paquetes. ¿Qué hacer en esos casos? Analicemos esos problemas uno a uno, aunque sea de manera somera.

- **Estructura de los búferes: compartidos o separados**

El espacio de memoria de los encaminadores puede ser compartido, para paquetes que lleguen por cualquier entrada, o puede repartirse a cada entrada, sólo para paquetes que lleguen por cada entrada.



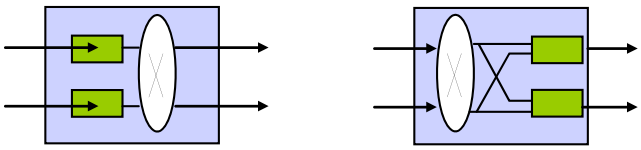
búferes compartidos búferes repartidos

Si el espacio de memoria es compartido, la capacidad de almacenamiento disponible se utiliza de manera más eficaz, ya que siempre hay sitio para un paquete hasta que se agota toda la memoria del encaminador; sin embargo, si lo repartimos entre los diferentes puertos de entrada del encaminador, puede que se llene una de las colas de entrada y no se pueda admitir más paquetes en esa entrada, aunque pudiera quedar sitio libre en otras colas. Pero, por otra parte, la gestión de una memoria común para todas las entradas es más compleja, tanto en la carga de nuevos paquetes como en la salida de los mismos: se necesita una cola multientrada, porque la carga de paquetes (de diferente tamaño) puede ser simultánea durante varios ciclos y desde varios puertos de entrada; también debe ser multisalida, para que un paquete que no puede continuar su viaje no bloquee a otros paquetes que sí podrían continuar. Como siempre, se trata de buscar un compromiso entre eficiencia y complejidad.

En muchos sistemas, los paquetes tienen diferentes niveles de prioridad; por ejemplo, los paquetes de control (en general, paquetes cortos) pueden tener prioridad sobre los paquetes de datos (más largos). En esos casos, los búferes se organizan en varias colas, una por cada clase de paquete.

- **Búferes en las entradas o en las salidas**

Los búferes de espera pueden asociarse tanto a los puertos de entrada como a los de salida. Si se colocan en la entrada, los paquetes pasan directamente al búfer de espera si su salida no está disponible o si hay paquetes esperando en los búferes de esa entrada. En cambio, si se colocan en la salida, primeramente se efectúa la operación de asignación de puerto de salida (encaminamiento) y luego se pasa, en su caso, al búfer de espera.



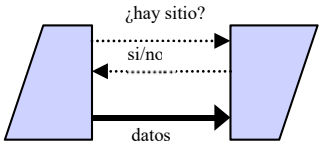
búferes en las entradas búferes en las salidas

Las ventajas y desventajas parecen claras. La gestión es más sencilla a la entrada, pero su uso no es tan eficiente (el primer paquete de una cola bloquea al resto de los paquetes, aunque éstos tengan camino libre). La gestión de los paquetes es más eficiente si se colocan a la salida, ya que ya se ha efectuado la asignación de salida para cuando se bloquea el paquete, pero su estructura es más compleja, ya que de nuevo deben ser colas multientrada.

- **¿Y si se llenan las colas de búferes?**

En momentos en que el tráfico de paquetes sea muy intenso puede que se sature la capacidad de almacenamiento de los búferes de los encaminadores y no se pueda admitir más paquetes. ¿Cómo se controla esa situación?

Lo más habitual es utilizar un protocolo de control entre encaminadores adyacentes (*link flow control*). Cuando hay que enviar un paquete de un encaminador a otro, primeramente se solicita permiso para la transmisión; cuando se recibe dicho permiso, se procede a transmitir los *flits* correspondientes. Si el receptor no dispone de sitio para el paquete que va a recibir, entonces no dará permiso de transmisión y ésta no se efectuará (por otra parte, la comunicación en modo WH requiere de un protocolo similar).



De esa manera se produce lo que se conoce como *back-pressure*: el no poder transmitir los paquetes hace que se llenen los búferes del encaminador y que éste transmita hacia atrás la imposibilidad de seguir recibiendo paquetes, con lo que, en un momento u otro, los nodos dejarán de inyectar más paquetes hasta que se recupere la posibilidad de seguir transmitiendo.

En la misma línea, en algunas máquinas se limita el número de paquetes que puede enviar un nodo, para evitar "inundaciones" de paquetes en determinados momentos. Cuando ya se ha recibido en destino el cupo de paquetes permitido, el procesador de destino da permiso al emisor para que pueda enviar más paquetes (una especie de *handshake* global). Una idea similar se aplica en el caso del conocido protocolo *token ring* (anillo con testigo), en el que un procesador no puede inyectar paquetes en la red salvo que disponga del testigo (*token*) que da permiso para ello. Dicho testigo es un mensaje de control especial que circula por toda la red de manera cíclica recorriendo todos los nodos de la misma.

Existen más técnicas para controlar el desbordamiento de la capacidad de almacenamiento de los encaminadores[1]. Por ejemplo, si llega un nuevo paquete a un encaminador y éste no dispone de sitio para

recibirlo por estar la correspondiente cola de búferes llena, siempre puede cogerse uno de los paquetes que está en la cola y "echarlo fuera", desviándolo a otro sitio, para dejar sitio al nuevo paquete. Está claro que utilizando esa estrategia siempre hay sitio para acoger paquetes, pero algunos paquetes van a verse penalizados al tener que efectuar recorridos más largos (con lo que su latencia será mayor). Para escoger el paquete que se va a desviar pueden utilizarse diferentes alternativas: al azar, por ejemplo, o tal vez en función del tiempo que el paquete lleve en la red, o de la distancia que le falte para llegar a destino. Por ejemplo, el encaminador conocido con el nombre de *chaos router* utilizó esta técnica.