Difusión de mensajes tree-multicast en redes de interconexión basadas en wormhole routing

Manuel Perez Malumbres y Jose Duato Marín

Universidad Politécnica de Valencia

Departamento de Ingeniería de Sistemas, Computadoresy Automática

e-mail: mperez@gap.upv.es, jduato@gap.upv.es

1. Introducción.

Los multicomputadores [1] son arquitecturas con múltiples procesadores y memoria local en cada procesador, que se comunican a través de una red de interconexión mediante paso de mensajes. En muchos casos, cada mensaje tien un único destino. En otros casos, el paso de mensajes a través de la red de interconexión utiliza difusiones *multidestino* y difusiones *broadcast*. En bastantes aplicaciones como simulaciones de redes de computadoras, de multiprocesadores, de circuitos, aplicaciones de cálculo intensivo, algoritmos numéricos paralelos, protocolos de invalidación de memorias cache en sistemas de memoria compartida, etc., es bastante usual enviar un mensaje hacia varios destinos (difusión de mensaje).

Cuando el conjunto de nodos destino sea el total de la red de interconexión, hablaremos de *difusión total* o *Broadcast*. Si el conjunto de nodos destino es mayor que uno y menor que el total, entonces tendremos una *difusión multidestino* o *Multicast*.

De aquí, podríamos decir que las comunicaciones multidestino son una generalización del resto, ya que las comunicaciones Unidestino (uno-a-uno) y Broadcast (uno-a-todos) son un caso particular de estas.

Actualmente, algunos multicomputadores comerciales como el *NCube-2* soportan Broadcast y una especie de multidestino restringido, donde el conjunto de destinos debe de formar un subcubo dentro de la topología [2].

En un principio, los primeros estudios sobre las comunicaciones multidestino [3][4] dieron lugar a varios modelos de grafos y algoritmos de encaminamiento. Sin embargo, en estos primeros trabajos no se tenía en cuenta el problema de los bloqueos en la red de interconexión.

Más tarde se presentaron tres protocolos multidestino [5]: *Multi-unicast*, *Resumable multicast* y *Restricted Branch multicast*, que intentan evitar el problema de los bloqueos. Sin embargo, estos protocolos estaban basados en técnicas de control de flujo del tipo Virtual Cut-through [6], y no se proponía ningún algoritmo de encaminamiento.

Posteriormente, se abordó el estudio de algoritmos de encaminamiento multidestino, libres de bloqueos, en multicomputadoras basados en Wormhole Routing [7][8] usando mecanismos de difusión de mensajes basados en caminos Hamiltonianos. En este trabajo se propusieron algoritmos de encaminamiento deterministas libres de bloqueos para mallas 2D: Dual-path y Multi-path.

Otras alternativas se estudiaron para la difusión total (broadcast) de mensajes [9] usando otro mecanismo de difusión llamado SDP (Spanning set of Dimensional-disjoint Path) que se demuestra libre de bloqueos y con buenos resultados frente a los expuestos anteriormente [7][8].

Ultimamente se han introducido nuevos algoritmos parcial y totalmente adaptativos para mallas 2D, que parecen ser libres de bloqueos: PM y FM[10]. Sin embargo, no existía ninguna metodología para desarrollar algoritmos de encaminamiento adaptativos multidestino que fuesen libres de bloqueos y fáciles de diseñar (véase la creciente complejidad de los algoritmos PM y FM). Por ello se desarrolló [11] una teoría que tiene en cuenta las nuevas dependencias de canales introducidas con el multidestino y que es una extensión de la ya existente para comunicaciones Unidestino (uno-a-uno) [12].

Derivado del trabajo realizado en [9] se ha desarrollado un nuevo mecanismo de difusión (parcial y total) de mensajes en redes Wormhole Routing. Este mecanismo, BRCP (Base-Routing-Conformed-Path) [13] es una adaptación del expuesto en [9] libre de bloqueos y con soporte para mensajes multidestino.

2. Mecanismos básicos en la difusión de mensajes

En este punto, vamos a hacer un repaso de los diferentes mecanismos propuestos para llevar a cabo la difusión de mensajes en redes de interconexión basadas en wormhole routing.

2.1. Tree-like Routing.

Se puede decir que un buen algoritmo de encaminamiento multidestino, además de ser libre de bloqueos, deberá llevar un mensaje multidestino a todos sus destinos en el menor tiempo posible, y utilizando el menor número de canales posible.

Para un conjunto determinado de destinos, el algoritmo de encaminamiento multidestino deberá llevar el mensaje, el mayor tiempo posible, a través del mismo camino . Cuando esto no se pueda mantener, se deberá ramificar el mensaje en varios, de forma que todos ellos alcancen los destinos por rutas diferentes. En esto se basa el *encaminamiento jerárquico* o *arbóreo* (*Tree-like routing*).

Un ejemplo de esto se muestra en la *figura 1*, donde el nodo *0* (0000) envía un mensaje de Broadcast en una topología 4-cubo. Por supuesto, se supone una red de interconexión basada en Wormhole Routing.

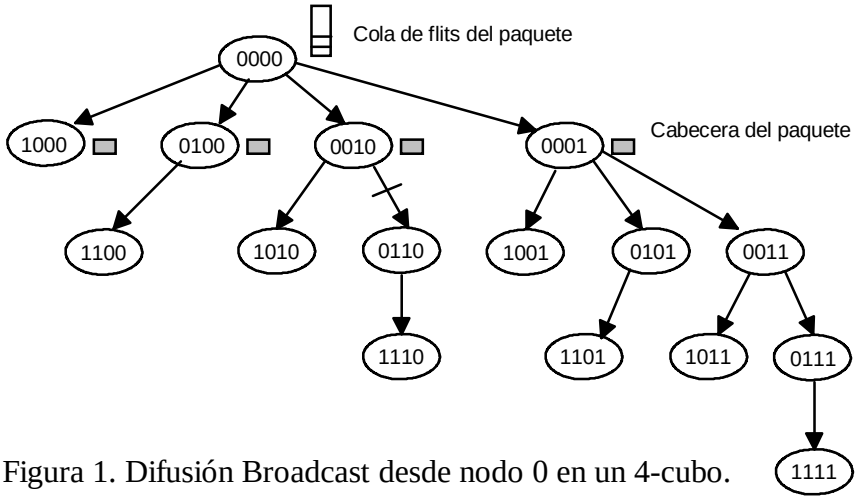


Figura 1. Difusión Broadcast desde nodo 0 en un 4-cubo.

Supongamos que en la figura, el canal *0010-0110* está ocupado. Cuando la cabecera avanza se bloqueará en *0010* esperando a que se libere dicho canal. En Wormhole Routing, cuando esto ocurre, el resto de flits también se bloquean. Aunque el resto de cabeceras replicadas en *0000* puedan avanzar hacia los destinos, el mensaje no puede avanzar hasta que se libere el canal anterior.

Sabiendo esto, el encaminamiento jerárquico (*Tree-like Routing*) incrementará notablemente la congestión de la red y degradará el rendimiento del multicomputador. Por otro lado es un mecanismo que incrementa notablemente la probabilidad de bloqueos en la red de interconexión.

Una solución razonable consistirá en prohibir a los nodos intermedios que realicen bifurcaciones, como ocurre en la *figura 1* en los nodos *0010*, *0001* y *0011*. Por tanto, se construirán un/os camino/s a seguir por el mensaje. Estos caminos no contendrán bifurcaciones y alcanzarán a todos los destinos. A este tipo de encaminamiento se le llama *Path-like Routing*.

2.2. Path-like Routing.

Un camino (*path*) multidestino consistirá en una sucesión de canales que comienza en el origen y alcanza a todos los destinos, siguiendo un determinado orden. Si sólo construimos un camino, este puede ser demasiado largo y quizás utilizará recursos innecesarios. Para solventar estos problemas podremos dividir el conjunto de destinos en varios subconjuntos disjuntos, de forma que desde el origen podamos enviar varios mensajes, cada uno con un subconjunto de destinos, que seguirán rutas independientes.

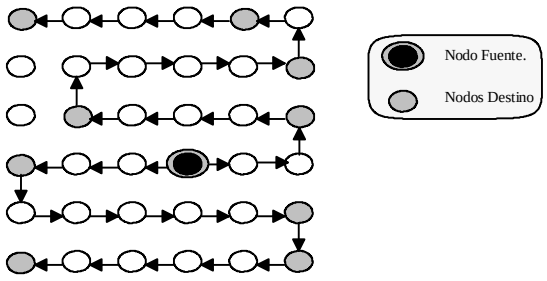


Figura 2. Dual-Path Routing en una malla 6x6.

En la figura 2 podemos ver un ejemplo de este mecanismo de difusión de mensajes.

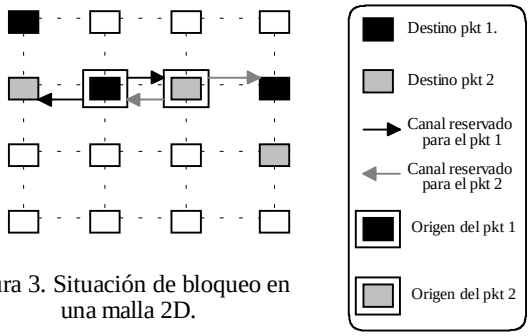


Figura 3. Situación de bloqueo en una malla 2D.

Todos los métodos estudiados hasta ahora para diseñar algoritmos de encaminamiento libres de bloqueos, no nos sirven cuando utilizamos multidestino. Con multidestino aparecen nuevas dependencias de canales que hay que analizar para decidir si un algoritmo es libre de bloqueos o no.

Por ejemplo, en comunicaciones *Unidestino* (uno-a-uno), el algoritmo de encaminamiento *X-First* para malla 2D es libre de bloqueos. Este algoritmo selecciona primero los canales horizontales y después los verticales para alcanzar un destino en la malla. Este mismo algoritmo de encaminamiento ya no es libre de bloqueos cuando usamos multidestino y *Path-like Routing*. En la figura 3 tenemos un ejemplo de un bloqueo entre dos mensajes multidestino.

2.3. Base-Routing-Conforming-Path (BRCP).

Este mecanismo de difusión toma un enfoque muy particular respecto a los anteriores. La idea es que los mensajes de difusión alcancen los destinos intentando seguir rutas mínimas para cada destino, cosa que el mecanismo *Path-based* no era capaz de conseguir. Además este mecanismo se adapta a cualquier tipo de encaminamiento básico utilizado para mensajes unidestino (uno-a-uno).

Fundamentalmente se proponen estrategias basadas en una difusión por etapas. Es decir, para enviar un mensaje de difusión (multicast o broadcast) primero tendremos que ordenar los destinos en función de su posición en la red de interconexión y del algoritmo de encaminamiento básico que se vaya a utilizar. A continuación, se envía un mensaje (unicast/multicast) que al llegar a su(s) destino(s) provoca que estos envíen, en una segunda fase de la difusión, mensajes multidestino a subconjuntos disjuntos de destinos. Esto se puede repetir en un número de fases que lógicamente debe ser pequeño.

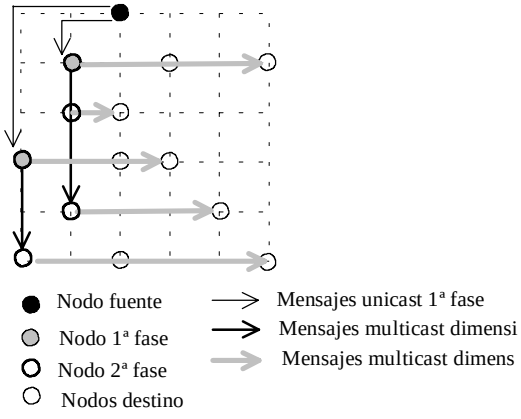


Figura 4. BRCP jerárquico en malla 2D. Encaminamiento e-c

En la figura 4 tenemos un ejemplo de una difusión multidestino siguiendo este mecanismo BRCP. Como se puede aprecia la difusión se completa tras realizar tres fases. La primera con mensajes unidestino, la segunda con worms multidestino que barren columnas, y en la última fase se generan mensajes multidestino para alcanzar a todos los destinos.

Este mecanismo también es libre de bloqueos si el algoritmo de encaminamiento base también lo es, como se demuestra en [13].

3. Un mecanismo alternativo: Multidestno arbóreo con Poda.

Hasta ahora se ha considerado el multidestino arbóreo (*tree-based multicast*) como un buen mecanismo para difusión de mensajes en redes de interconexión tipo Store&Forward. En cambio, con la llegada del control de flujo wormhole, este mecanismo se hace muy sensible a los bloqueos en la Red de Interconexión (RI). De ahí, que en la práctica, se hayan estudiado otros mecanismos de difusión como el *path-based multidestino* [7] en el que un mensaje multidestino va visitando los destinos de forma ordenada y secuencial. Con este mecanismo de encaminamiento, es más fácil evitar los bloqueos en la RI. Con una adecuada ordenación de los destinos a visitar y un algoritmo de encaminamiento libre de bloqueos, se garantiza la ausencia de los mismos.

Lo que realmente hace poco eficiente el *path-based*, es la fase de preparación de cada mensaje multidestino, ya que, en ella, se debe realizar una ordenación de los destinos con un coste software de al menos O(n*log n). Este coste hace que la latencia total de cada mensaje se incremente en uno o más ordenes de magnitud.

Por otro lado, al ordenar los destinos e ir visitándolos secuencialmente, el encamina miento del mensaje no es mínimo para todos los destinos, por lo que sus prestaciones no serán todo lo óptimas que quisiéramos.

Por ello, hemos vuelto a considerar el mecanismo multidestino arbóreo, el cual no necesita una ordenación previa de destinos. Además, todos los mensajes multidestino alcanzan a todos sus destinos siguiendo ruta mínima. El problema que deberemos de solucionar son los bloqueos que este mecanismo introduce.

3.1. Mecanismo tree-based multicast

Cada mensaje contiene varias direcciones de nudo o cabeceras, tantas como destinos tenga el mensaje a enviar. En este mecanismo, cada cabecera del mensaje es encaminada de forma independiente a las demás en cada nodo que atraviesa, pudiendo decidir la ruta que más le interese (seguir las rutas que han abierto las cabeceras anteriores o abrir una nueva para alcanzar su destino). De esta manera un mensaje multidestino de este tipo podrá abrir tantas ramas como necesite en su avance hacia los destinos. Como veremos después, esto no aumenta la complejidad de conmutador existente en cada nudo, ya que no se requiere conectar ninguna entrada con varias salidas simultáneamente.

El formato del mensaje multidestino que vamos a usar para este mecanismo será:

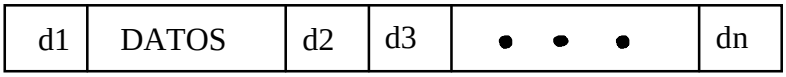


Figura 5. Formato de un mensaje multidestino arbóreo con poda.

Donde el primer flit corresponde a la primera cabecera (d1) a encaminar, a continuación los flits de datos del mensaje, y por último el resto de cabeceras con los destinos a alcanzar. Los flits de datos se deben almacenar en cada nodo intermedio (en un buffer auxiliar que debe tener asociado a cada canal de entrada) por donde pase el mensaje con objeto de mantener el formato del mensaje cuando hay bifurcaciones. Así, cuando una cabecera (p.e. d6) se encamina en un nodo, visitado previamente por otra cabecera (p.e. d5) del mensaje, pueden suceder dos cosas:

(a) Si *d6* abre una nueva ruta, es decir, no sigue el camino establecido por alguna de las cabeceras previas (*d1..d5*), ese nodo debe inyectar los flits de datos del mensaje multidestino que tiene almacenados detrás de esta cabecera, y que serán almacenados en el próximo nodo de esta nueva ruta.

(b) Si al encaminarse, decide seguir una ruta previamente establecida por otra cabecera, entonces, el flit de esa cabecera deberá cruzar el nodo por dicha ruta. Por supuesto, sin enviar a continuación los flits de datos, ya que estos se enviaron tras la primera cabecera que siguió esta ruta.

De este modo, el conmutador requiere transmitir cada flit que llega por una entrada a una sola salida. En la figura 6 se ve un ejemplo de la evolución de un mensaje multidestino con este mecanismo. Se utiliza un encaminamiento básico determinista tipo *X-First*.

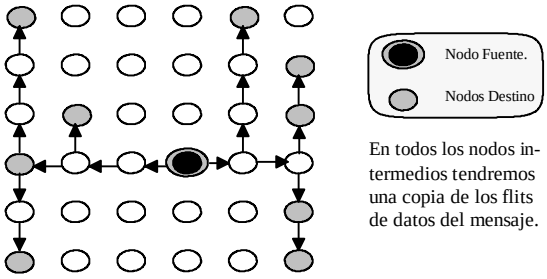


Figura 5 Tree-mcast con poda en malla 2D.

El algoritmo de encaminamiento básico (heredado de los mensajes unidestino) primero encamina por las filas (eje *x*) y después por las columnas (eje *y*). Como podemos apreciar, alcanzamos cada destino del mensaje multidestino con el mismo algoritmo de encaminamiento que usaríamos si enviáramos un mensaje unidestino a cada uno de ellos. Esto es de gran utilidad para el diseño de circuitos de encaminamiento, pues el mismo algoritmo sirve para todos los casos.

En la figura 5 podemos apreciar que en el nodo origen hay una bifurcación. Una rama alcanzará los destinos que tiene a su izquierda y la otra a los que se encuentren a su derecha. Por la rama que va a la izquierda viajan las cabeceras para esos destinos y los flits de datos siguiendo el formato que hemos visto. Los flits de datos se almacenan internamente en cada nodo intermedio, con el objeto de prever una posible ramificación en cualquiera de ellos, y así poder construir la nueva rama que tras la cabecera debe llevar los flits de datos.

3.2 Mecanismo libre de bloqueos.

Este mecanismo nos conduce a un aumento de las ramificaciones en un mensaje multidestino, haciendo que la probabilidad de bloqueo sea elevada (se crean nuevas dependencias que el algoritmo de encaminamiento no puede eliminar).

Una posible solución al bloqueo es controlar la ramificación de los mensajes multidestino mediante un mecanismo de poda. Así cada vez que se bloquea un mensaje multidestino en un determinado nodo, se procede a la poda de todas las ramificaciones que en ese momento tiene dicho mensaje multidestino en el nodo que generó la rama que no puede

avanzar. El propio mecanismo de control de flujo notifica a este nudo que la rama no puede avanzar. Así, las ramas recién podadas puedan avanzar libremente y liberar canales que puedan bloquear a otros mensajes.

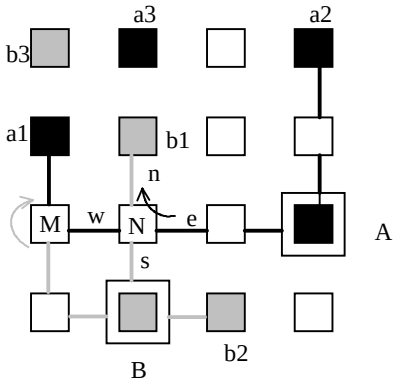


Figura 6: Los nodos A y B envían dos mensajes multicast de 3 destinos. Los mensajes se bloquean mutuamente en los nodos M y N.

Por ejemplo, en la figura 6 se muestran dos mensajes multidestino que se cruzan en el nodo N. La cabecera *a1* cruza N de este a oeste reservando los canales w(est) y e(ast), mientras que la primera cabecera de B, *b1*, cruza N de sur a norte, reservando a su vez los canales s(outh) y n(orth).

Cuando llega la cabecera *a3* al nodo N, se encamina y se encuentra el canal ocupado por otro mensaje (B) bloqueándose en N. A su vez, cuando la cabecera *b3* llega a M, al encaminarse, descubrirá que el canal que necesita para alcanzar a su destino esta ocupado por otro mensaje (A), bloqueándose también. Por tanto hemos llegado a un bloqueo mutuo entre los mensajes A y B.

Para deshacerlo, se procede a la poda en N. Es decir, cuando el nodo N encamina a3 y ve que se bloquea, poda todas las ramas del mensaje A que se han abierto en N. En este caso sólo tenemos una rama: la que va hacia el oeste, cuando se hace la poda, la rama(s) podada(s) avanzará hacia su destino y en un futuro liberará ese canal que está bloqueando al mensaje B.

El mecanismo de poda es capaz de eliminar los bloqueos que se producen por el efecto de las ramificaciones. Actualmente estamos desarrollando la base teórica para asegurar la ausencia de bloqueos.

En el ejemplo, hemos utilizado una poda total en el nodo N (todas las ramas del mensaje se cortan), sin embargo no es necesaria una poda total para garantizar la ausencia de bloqueos.

3.3 Limitaciones de este mecanismo.

La limitación obvia es el tamaño de los datos de los mensajes multidestino. Este debe ser pequeño (unos pocos flits) ya que en cada canal de entrada de cada nodo debemos disponer un buffer capaz de albergar todos los flits de datos de un mensaje multidestino.

Sin embargo, puede ser muy útil en entornos particulares como los protocolos de invalidación de memoria cache en multiprocesadores con memoria compartida, donde determinados mensajes multidestino utilizados en los protocolos de coherencia se adaptan a las características de este mecanismo.

3.4. Evaluación de prestaciones.

Se ha desarrollado un simulador de redes de interconexión a nivel de flit, donde se ha implementado el mecanismo *multidestino arbóreo con poda*. En este simulador se han lanzado varias simulaciones cuyo objetivo era averiguar si aportaba ventajas usar este mecanismo frente al multidestino simulado (un mensaje unidestino por cada destino).

Se han lanzado simulaciones sobre distintas topologías: mallas 2D, toros 2D y 3D, con distintos tamaños. Los mensajes multidestino tienen un flit de datos (valor típico en los protcolos de invalidación). El número de destinos de cada mensaje oscilará entre 4 y 25 destinos. Se han utilizado algoritmos de encaminamiento deterministas (siempre se sigue la misma ruta entre cada par origen-destino). El tráfico global está compuesto sólo por mensajes multidestino.

A continuación se muestran algunas gráficas que comparan el *multidestino arbóreo con poda* con el multidestino simulado (*pseudo-multicast*) en diferentes condiciones de tráfico y con mensajes multidestino con N destinos de media.

Malla 2D (8x8)

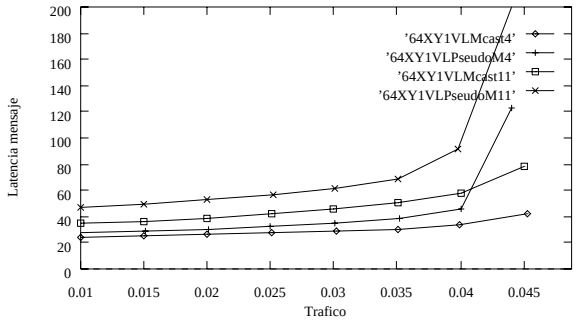


Figura (a)

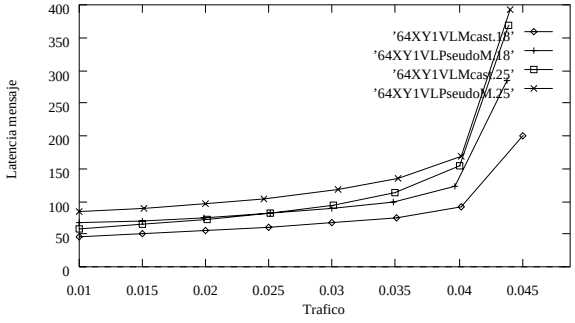


Figura (b)

Toro 2D (8x8)

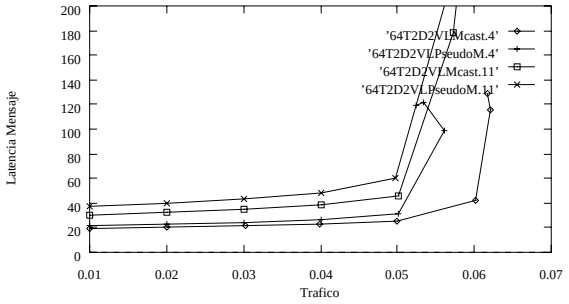


Figura (c)

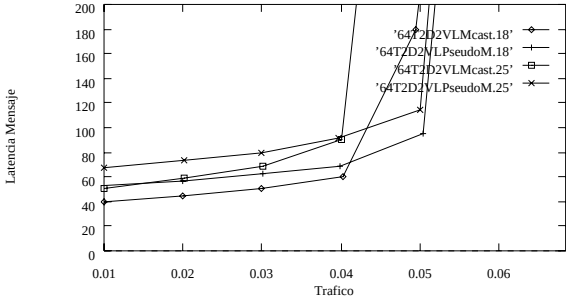


Figura (d)

Toro 3D (8x8x8)

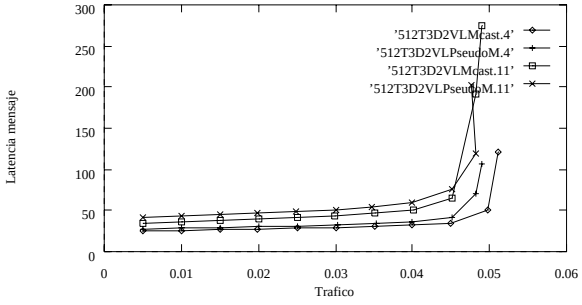
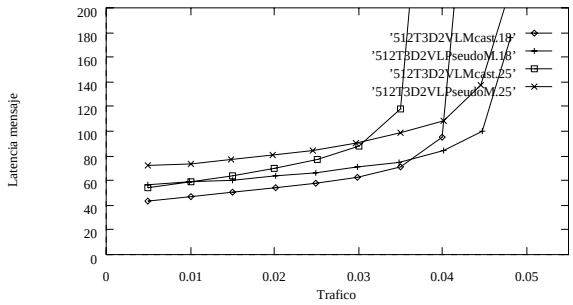


Figura (e)



Se puede observar que en la malla2D, para pocos destinos, se obtienen mejoras de hasta el 25% respecto al pseudo-multicast (multidestino simulado con mensajes unidestino). El algoritmo de encaminamiento es determinista y no se utilizan canales virtuales. En los toros2D las mejoras no son tan fuertes como en las mallas. El algoritmo de encaminamiento sigue siendo determinista peor ahora usamos 2 canales virtuales por canal físico. Algo similar ocurre con el toro3D.

En los toros no se gana mucho. Esto puede ser debido a que usamos canales virtuales. Con los canales virtuales se porporcionan más rutas alternativas para alcanzar un destino desde un nodo determinado, por lo que los mensajes multidestino se ramificarán bastante más y por tanto se producirán más podas. Cosa que frena las prestaciones del mecanismo tree-multicast. Para mejorar esto se deben estudiar podas selectivas para que no grave en exceso la caída de prestaciones.

3.5 Conclusiones.

Se ha descrito un nuevo mecanismo para el transporte de mensajes multidestino en redes de interconexión. Este mecanismo intenta mejorar algunos aspectos de los mecanismos ya estudiados, volviendo a una variante del mecanismo tree-multicast.

Entre otras cosas, este mecanismo:

- No necesita ningún tipo de ordenación previa de los destinos.

- Alcanza a cada destino siguiendo una ruta mínima (si el algoritmo de encaminamiento base es de ruta mínima, claro está).

- Se adapta a cualquier topología

- Puede usar cualquier algoritmo de encaminamiento (siempre y cuando sea libre de bloqueos).

- La libertad de bloqueos se consigue gracias a un dispositivo de poda.

En cuanto a prestaciones, se observa que el *tree-multicast* mejora la latencia en todos los casos, excepto cuando la red se acerca al punto de saturación y el número de destinos es elevado. Como el empeoramiento cuando la carga es elevada se debe a la poda incontrolada, pensamos analizar el comportamiento utilizando podas controladas.

Actualmente se está desarrollando la base teórica que demuestra la ausencia de bloqueos en redes de interconexión usando el mecanismo *multidestino arbóreo con poda*. También, se están estudiando mecanismos de poda selectiva que mejorarán las prestaciones, sobre todo cuando la carga de la red es elevada. Por otro lado, se ve interesante el estudiar el comportamiento de este mecanismo junto a los protocolos de coherencia en multiprocesadores con memoria compartida en situaciones de carga real.

4. Referencias.

[1] W.C.Athas y C.L.Seitz, "*Multicomputers: Message-passing concurrent computers*". IEEE Comput. Mag. vol. 21, no. 8, pp. 9-24, Agosto 1988.

[2] NCUBE Company, NCUBE 6400 Processor manual, 1990.

[3] Y. Lan, A.H. Esfahanian y L.M. Ni, "*Multicast in hypercube multiprocessors*". Journal of parallel and distributed computing, pp. 30-41, Jan. 1990.

[4] X. Lin y L.M. Ni, "*Multicast communication in multicomputer networks*". Proc. Int. Conf. on Parallel Processing, pp. III-114-III-118, Aug. 1990.

[5] G.T. Byrd, N.P. Saraiya y B.A. Delagi, "*Multicast communication in multiprocessor systems*". Proc. Int. Conf. of Parallel Processing, pp. I-196-I-200, 1989.

[6] P.Kermani y L.Kleinrock, "*Virtual Cut-through: a new computer communication switching technique*", Comput. Networks, vol. 3, pp. 267-286, 1979

[7] X. Lin y L.M. Ni, "*Deadlock-free Multicast wormhole routing in multicomputer networks*". Proc. Int. Symp. on Computer Arquitecture, May 1991.

[8] X.Lin, P.K. McKinley y L.M. Ni, "*Performance evaluation of Multicast wormhole routing in 2D-Mesh Multicomputers*". Proc. Int. Conf. on Parallel Processing, 1991, pp. I-435-I-442.

[9] K.Panda y S.Singal, "*Broadcasting in k-ary n-cube Networks using Multidestination Worms".* Proc. International Conference on Parallel Processing, 1994.

[10] X.Lin, P.K. McKinley y A.H. Esfahanian, "*Adaptive multicast wormhole routing in 2D-Mesh Multicomputers*". Proc. Parallel Architectures Languages Europe 93, Junio 1993.

[11] J. Duato, "*On the design of deadlock-free adaptive multicast routing algorithms*". Parallel Processing Letters Vol. 3, No. 4, 1993.

[12] J. Duato, "*A new theory of deadlock-free adaptive routing in wormhole networks*". IEEE Trans. Parallel and Distributed Systems. 1993.

[13] K.Panda, S.Singal y P.Prabhakaran, "*Multidestination Message Passing Mechanism Conforming to Base Wormhole Routing Scheme".* Tech. Report OSU-CISRC-6/94-TR33, Dept. of CIS, The Ohio State University, 1994.