ALGORITMOS DE ENCAMINAMIENTO MULTIDESTINO SOBRE REDES DE INTERCONEXION BASADAS EN WORMHOLE ROUTING.

Manuel Pérez Malumbres

Departamento de Ingeniería de Sistemas, Computadores y Automática

Universidad Politécnica de Valencia

46071 Valencia

E-mail: mperez@aii.upv.es

Resumen

Esta parte trata de dar una visión general sobre los mecanismos de difusión de mensajes en una red de interconexión. Iremos repasando los distintos algoritmos que se han propuesto para esta forma de comunicaciones a lo largo de los últimos años. También revisaremos una teoría para el diseño de algoritmos de encaminamiento multidestino (o multicast) adaptativos libres de bloqueo, que permite diseñar fácilmente algoritmos adaptativos multidestino garantizando la ausencia de bloqueos.

Por último, se hablará de las herramientas de simulación utilizadas para la obtención de información acerca del rendimiento de estos algoritmos y de otros nuevos que se propongan. Se presentan gráficas comparativas entre los distintos algoritmos estudiados.

1. Introducción.

Otra forma de realizar el paso de mensajes a través de la red de interconexión es utilizar difusiones *multidestino* y difusiones *broadcast*. En bastantes aplicaciones como simulaciones de redes de computadores, de multiprocesadores, de circuitos, aplicaciones de cálculo intensivo, algoritmos numéricos paralelos, etc., es bastante usual enviar un mensaje que sea recibido por 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 multicast restringido, donde el conjunto de destinos debe de formar un subcubo dentro de la topología [1].

En un principio, los primeros estudios sobre las comunicaciones multicast [2][3] 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 multicast [4]: *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, y no se proponía ningún algoritmo de encaminamiento.

Posteriormente, se abordó el estudio de algoritmos de encaminamiento multicast, libres de bloqueos, en multicomputadores basados en Wormhole Routing [5][6]. En este trabajo se proponen algoritmos de encaminamiento deterministas libres de bloqueos para mallas 2D: Dual-path y Multi-path.

Ultimamente se han introducido nuevos algoritmos parcial y totalmente adaptativos para mallas 2D, que parecen ser libres de bloqueos: PM y FM[7]. Sin embargo, no existía ninguna metodología para desarrollar algoritmos de encaminamiento adaptativos multicast que fuesen libres de bloqueos y fáciles de diseñar (véase la creciente complejidad de los algoritmos PM y FM). Por ello en [8] se desarrolló una teoría que tiene en cuenta las nuevas dependencias de canales introducidas con el multicast y que es una extensión de la ya existente para comunicaciones Unicast (uno-a-uno) [9].

2. Tree-like Routing.

Se puede decir que un buen algoritmo de encaminamiento multicast, además de ser libre de bloqueos, deberá llevar un mensaje multicast 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 multicast 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.

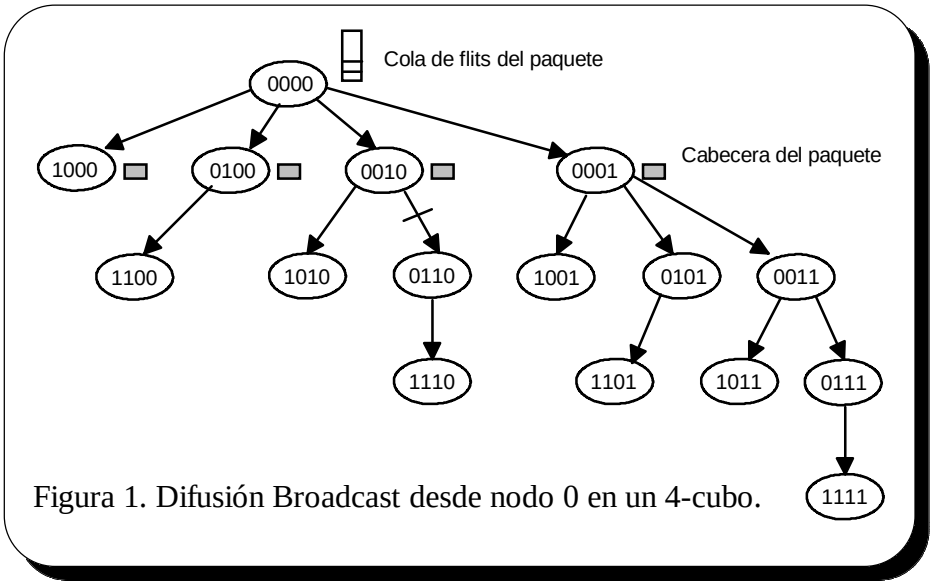

Figura 1. Difusión Broadcast desde nodo 0 en un 4-cubo.

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 incrementará notablemente la congestión de la red y degradará el rendimiento del multicomputador. Una solución razonable consistirá en prohibir a los nodos intermedios que repliquen el mensaje en varios canales de salida, 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*.

3. Path-like Routing.

Un camino multicast 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 podremos enviar varios mensajes, cada uno con un subconjunto de destinos, que seguirán rutas independientes.

Parece claro y demostrable que si utilizamos encaminamiento jerárquico, la probabilidad de bloqueo de un mensaje es mayor que si utilizamos *Path-like Routing* [2].

Ahora sabemos que un encaminamiento jerárquico es negativo para la red, y que una solución es usar encaminamiento *Path-like Routing*, pero aun tenemos el problema de los bloqueos por resolver.

Todo lo comentado en secciones anteriores acerca de los métodos para diseñar algoritmos de encaminamiento libres de bloqueos, no nos sirve cuando utilizamos multicast. Con multicast aparecen nuevas dependencias de canales que hay que analizar para decidir si un algoritmo es libre de bloqueos o no.

Por ejemplo, en comunicaciones *Unicast* (uno-a-uno), el algoritmo de encaminamiento *X-First* para malla 2D era libre de bloqueos. Este algoritmo seleccionaba primero los canales horizontales y después los verticales para alcanzar un destino en la malla.

El mismo algoritmo de encaminamiento en comunicaciones multicast no es libre de bloqueos (ver *figura 2*).

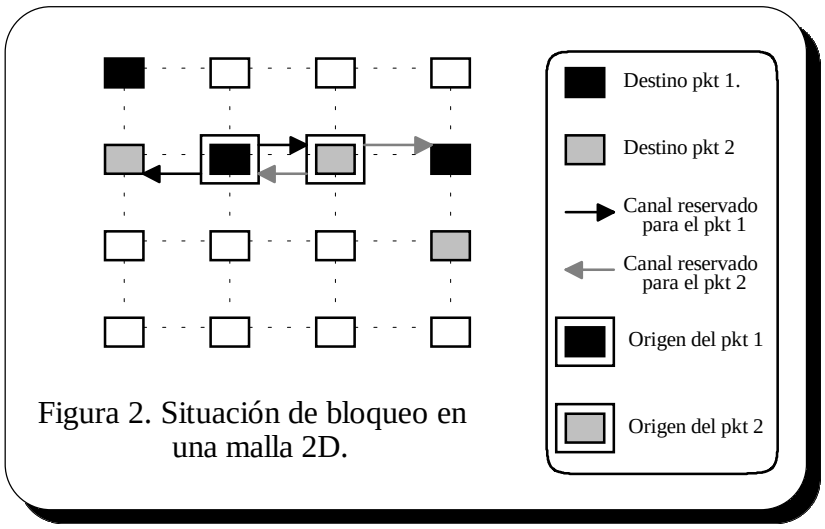

Figura 2. Situación de bloqueo en una malla 2D.

Se proponen una serie de algoritmos de encaminamiento multicast, libres de bloqueo, para mallas 2D [6] que utilizan un encaminamiento tipo *Path-like*: Double-Channel, Dual-Path y Multi-Path.

3.1 Double-Channel.

Consiste en una modificación del algoritmo de encaminamiento *X-First*. Para romper la condición de bloqueos (dependencias cíclicas de canales), se duplican los canales de la red de interconexión. Con esto, podríamos considerar que la red de interconexión puede ser dividida en cuatro subredes virtuales: $N_{+X,+Y}$, $N_{-X,+Y}$, $N_{-X,-Y}$, $N_{+X,-Y}$, de forma que la red $N_{+X,+Y}$ contiene todos los canales unidireccionales que conectan los nodos *[(i,j), (i+1,j)]* y *[(i,j),(i,j+1)]*. En la *figura 3* se muestra la división de una malla 2D en las cuatro redes virtuales.

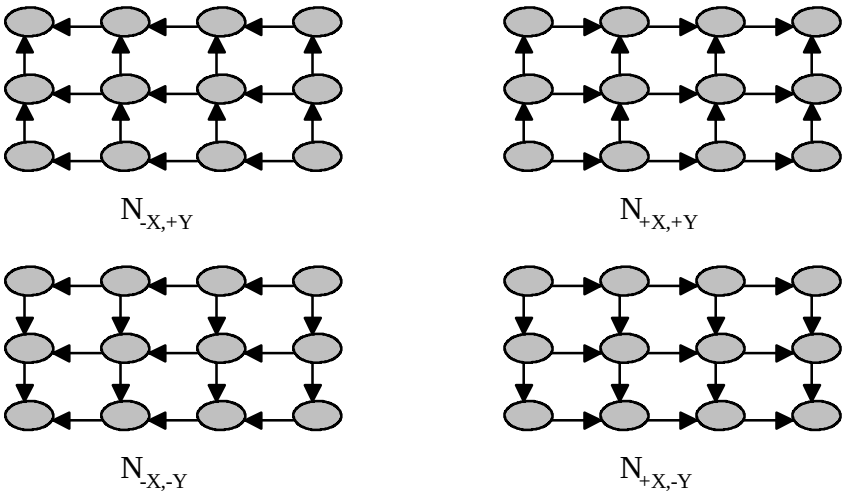

Figura 3. Partición de una malla 2D usando Double-Channel.

El conjunto de destinos de un mensaje multicast podrá ser dividido como máximo en cuatro subconjuntos : $D_{+X,+Y}$, $D_{+X,-Y}$, $D_{-X,+Y}$ y $D_{-X,-Y}$, de acuerdo a la posición de los destinos respecto al nodo fuente u_0. Así, los nodos destino que se encuentren por encima y a la derecha de u_0 se incluirán en el subconjunto $D_{+X,+Y}$.

Ejemplo: Sea una malla 6x6, donde el nodo u_0 de coordenadas (x_0=3, y_0=2) envía un mensaje multicast hacia D = { (0,0), (0,2), (0,5), (1,3), (4,5), (5,0), (5,1), (5,3), (5,4) }.

Intentamos dividir *D* en subconjuntos de destinos, siguiendo las siguientes acotaciones:

$$D_{+X,+Y} = \{(x,y) / \quad x > x_0,\ y \geq y_0\} = \{(4,5),(5,3),(5,4)\}$$

$$D_{-X,+Y} = \{(x,y) / \quad x \leq x_0,\ y > y_0\} = \{(0,5),(1,3)\}$$

$$D_{-X,-Y} = \{(x,y) / \quad x < x_0,\ y \leq y_0\} = \{(0,0),(0,2)\}$$

$$D_{+X,-Y} = \{(x,y) / \quad x \geq x_0,\ y < y_0\} = \{(5,0),(5,1)\}$$

En la *figura 4* se muestra la generación de cuatro mensajes multicast que circularán por cada una de las redes virtuales definidas.

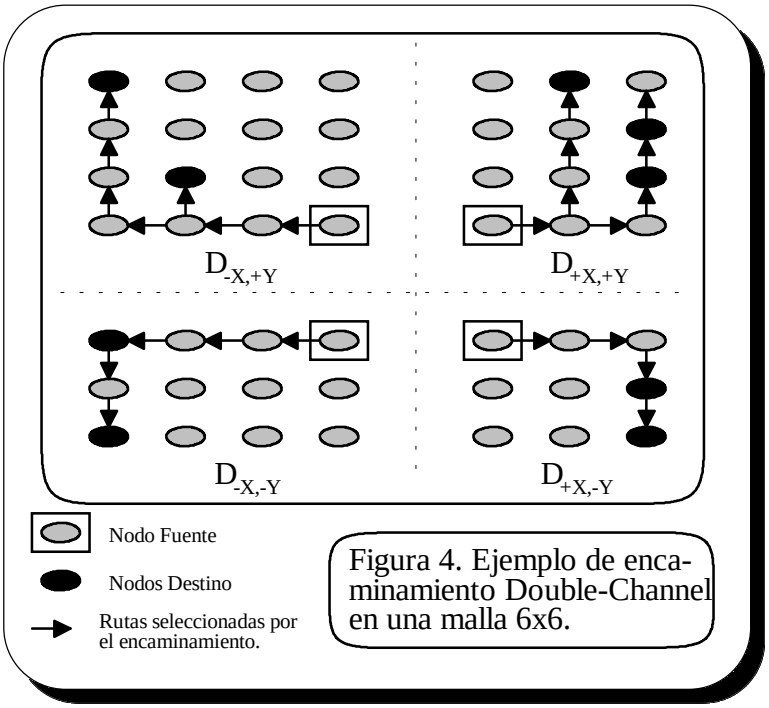

Figura 4. Ejemplo de encaminamiento Double-Channel en una malla 6x6.

Con esta aproximación evitamos los bloqueos, pero, sin embargo, hemos tenido que aumentar el número de canales entre nodos. Esto puede hacerse de forma equivalente y sencilla con el uso de canales virtuales. En este algoritmo todavía permitimos que nodos intermedios puedan replicar mensajes haciendo complejo el diseño y la señalización multicast en el router de cada nodo. Por ello los siguientes algoritmos evitan duplicar los canales de la red de interconexión y también evitan la réplica de mensajes en nodos intermedios.

Estos algoritmos: Dual-Path y Multi-Path, dividen la red de interconexión en subredes (como en el caso anterior) basándose en un camino Hamiltoniano. El camino Hamiltoniano visita a todos los nodos de la red una y solo una vez. En una malla 2D existen muchos caminos Hamiltonianos.

Para construir un camino Hamiltoniano utilizaremos una función de etiquetado *l(i)* que asignará etiquetas a cada nodo de la red. El camino recorre nodos con valores de etiquetas crecientes. La etiqueta del primer nodo del camino Hamiltoniano será *0*, y la del último será *N-1* (donde *N* es el número de nodos de la malla).

La función de etiquetado *l* que utilizaremos para una malla 2D de *m x n* dimensiones será la siguiente:

$$l\ (x\ ,y) = \left\{ \begin{array}{c} y * n + x,\ \text{si y es par} \\ y * n + n - x - 1,\ \text{si y impar} \end{array} \right\}$$

El etiquetado de nodos según la función *l*, divide la red de interconexión en dos subredes:

- La subred con canales ascendentes, que contendrá todos los canales que conectan nodos con etiquetas bajas a nodos con etiquetas altas.
- La subred con canales descendentes, que incluye todos los canales que conectan nodos de etiquetas altas con nodos de etiquetas bajas.

En la *figura 5* podemos ver un etiquetado de una malla 2D de *4 x 4* elementos y su división en las dos subredes que hemos visto.

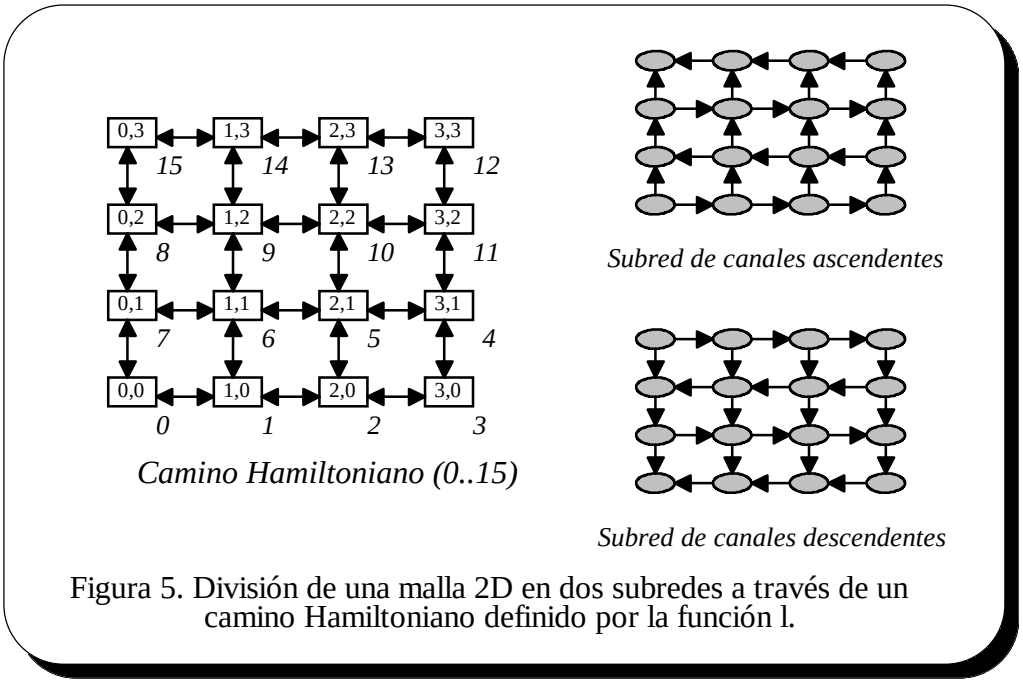

Figura 5. División de una malla 2D en dos subredes a través de un camino Hamiltoniano definido por la función l.

Para buscar un algoritmo de encaminamiento libre de bloqueos para este tipo de redes, se deben de utilizar las subredes encontradas de forma que no aparezcan dependencias cíclicas de canales.

Para ello el algoritmo de encaminamiento debe de ser cuidadoso a la hora de seleccionar los canales de salida. Un algoritmo de encaminamiento basado en el camino Hamiltoniano sería el siguiente:

R: NxN -> N, función de encaminamiento, se define como *R(u,v) = w*, de tal forma que: Si *l(u) < l(v)* entonces *w* será el vecino de *u* que tenga la mayor etiqueta, y si *l(u) > l(v)*, entonces *w* será el vecino de *u* que tenga la menor etiqueta.

Se demuestra que este algoritmo de encaminamiento multicast es libre de bloqueos [5]. Además será el utilizado tanto por Dual-Path como por Multi-Path.

4.2 Dual-Path.

Este algoritmo divide el conjunto de destinos *D* de un mensaje multicast en dos subconjuntos D_H y D_L, donde cada nodo perteneciente a D_H tendrá una etiqueta mayor que el nodo fuente u_0, y cada nodo de D_L tendrá una etiqueta menor que u_0.

Como hemos visto en Double-Channel, cuando u_0 envía un mensaje multicast a un determinado conjunto de destinos, se procede a intentar dividir ese conjunto de destinos. En Dual-Path, el algoritmo de división y sería el siguiente:

Entrada: Conjunto de destinos D y dirección local u_0.

Salida: Dos listas ordenadas de nodos: D_H y D_L.

Procedimiento:

1. Divide D en dos conjuntos D_H y D_L, de la siguiente forma:

$$D_L = \{n \in D / \quad l(n) < l(u_0)\}$$

$$D_H = \{n \in D / \quad l(n) > l(u_0)\}$$

2. Ordena los conjuntos D_H y D_L según su etiqueta. En D_H el orden será ascendente, y en D_L descendente.

3. Construye dos mensajes. Uno llevará en su cabecera D_H y el otro D_L

El algoritmo de encaminamiento que hemos introducido anteriormente se puede esquematizar de la siguiente manera:

Entrada: Mensaje con una lista de destinos ordenada $D_M = \{d_1, d_2, ..., d_k\}$ y una dirección local w.

Procedimiento:

1. Si $w = d_1$ entonces $D'_M = D_M - \{d_1\}$, enviandose copia del mensaje a *w.*
   Sino $D'_M = D_M$.
2. Si $D'_M = \varnothing$ entonces fin de transmisión de mensaje.
3. Sea *d* el primer nodo de D'_M y $w' = R(w,d)$.
4. El mensaje es enviado a *w'* con la lista de destinos D'_M en su cabecera.

Por construcción, a la hora de partir la red en dos subredes, el algoritmo de encaminamiento es libre de bloqueos.

3.2 Multi-Path.

Dual-Path obtendrá buenas prestaciones en función de donde estén colocados los destinos. En la *figura 6* se muestra un ejemplo donde se puede apreciar un caso desfavorable.

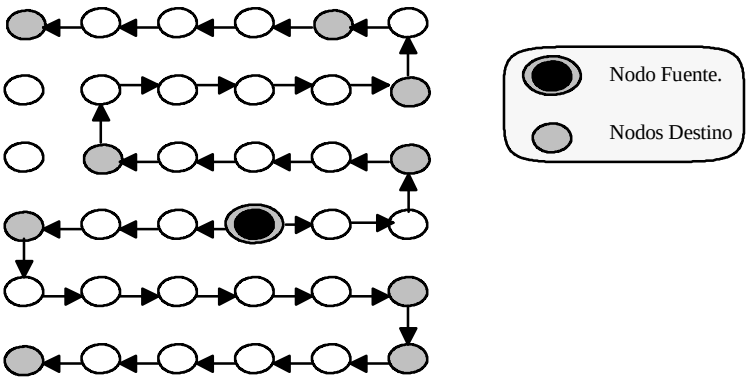

Figura 6. Dual-Path Routing en una malla 6x6.

El nodo (3,2) envía un mensaje multicast hacia 9 destinos colocados como muestra la figura. Se utilizan 33 canales (18 en la red de canales ascendentes y 15 en la otra). La máxima distancia del nodo fuente a un destino es de 18 saltos.

Para reducir el número de recursos y las distancias medias desde el nodo fuente, se utilizará el encaminamiento Multi-Path. La única diferencia respecto al algoritmo anterior está en la fase de preparación del mensaje, donde se decide cuantas rutas independientes se escogen para alcanzar a todos los destinos. El algoritmo de preparación del mensaje es el siguiente:

Entrada: Conjunto de destinos D y una dirección local $u_0 = (x_{0,}\, y_0)$.
Salida: Cuatro conjuntos de destinos ordenados.
Procedimiento:

1. Se divide D en D_H y D_L, de la misma forma que en Dual-Path.
2. Se ordenan los subconjuntos D_H y D_L.
3. Suponiendo que $v_1 = (x_{1,}\, y_1)$ y $v_2 = (x_{2,}\, y_2)$ son los nodos vecinos de u_0, se divide D_H en dos subconjuntos, D_{H1} y D_{H2} :

$$D_{H1} = \{(x,y)/ \quad x \le x_1 \text{ si } x_1 < x_2;\ x \ge x_1 \text{ si } x_1 > x_2\}$$
$$D_{H2} = \{(x,y)/ \quad x \le x_2 \text{ si } x_2 < x_1;\ x \ge x_2 \text{ si } x_2 > x_1\}$$

Se construyen dos mensajes, uno de ellos llevará en su cabecera la lista de destinos D_{H1}, y el otro la lista D_{H2}.

4. De forma análoga, se divide D_L en D_{L1} y D_{L2}.

Utilizando el mismo ejemplo que en Dual-Path, el nodo de coordenadas (3,2) se dispone a enviar un mensaje multicast hacia nueve destinos (los mismos que en la *figura 6*). Si aplicamos el algoritmo Multi-Path que acabamos de ver, primero construiríamos los conjuntos D_H y D_L :

$$D_H = \{(5,3),(1,3),(5,4),(4,5),(0,5)\}$$
$$D_L = \{(0,2),(5,1),(5,0),(0,0)\}$$

A continuación dividiríamos cada uno de estos conjuntos en otros dos más, obteniendo cuatro subconjuntos del conjunto original de destinos.

$$D_{H1} = \{(5,3),(5,4),(4,5)\}$$
$$D_{H2} = \{(1,3),(0,5)\}$$
$$D_{L1} = \{(5,1),(5,0)\}$$
$$D_{L2} = \{(0,2),(0,0)\}$$

Para este multicast se requieren un total de 20 canales, y la máxima distancia entre el nodo fuente y los destinos es 6 (véase la *figura 7*). Como se puede apreciar, Multi-Path es mejor que Dual-Path en términos de tráfico, recursos y distancia media a los destinos.

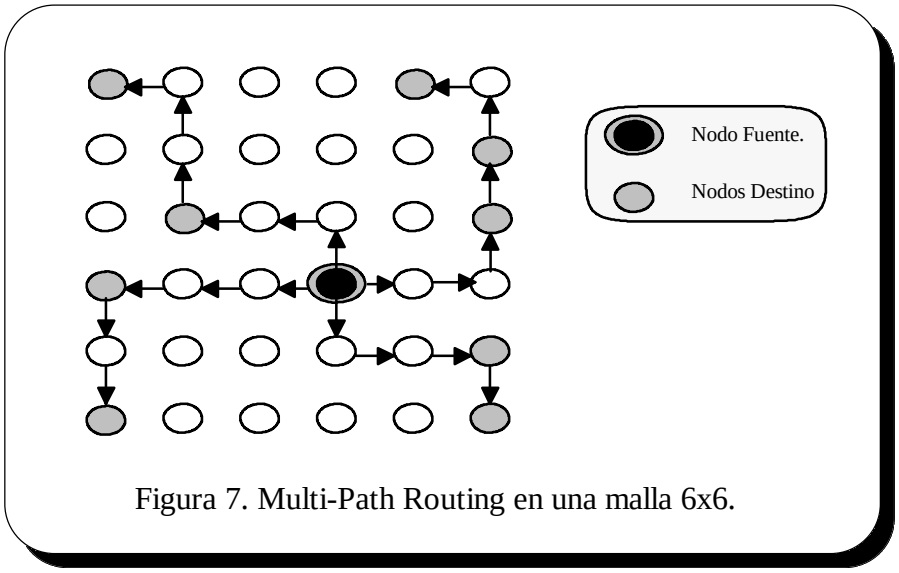

Figura 7. Multi-Path Routing en una malla 6x6.

5. Algoritmos adaptativos para multicast

Estos algoritmos de encaminamiento multicast, que acabamos de describir, son atractivos ya que son libres de bloqueos y pueden comportarse bien en una red poco cargada. Sería más interesante aún hacer que fuesen adaptativos, con lo que se acoplarían mejor a las condiciones dinámicas de una red. Por esto aparecen nuevos algoritmos de encaminamiento adaptativos como: PM (parcialmente adaptativo) y FM (totalmente adaptativo).

5.1 Algoritmo PM.

Este algoritmo de encaminamiento es mínimo, parcialmente adaptativo y no requiere el uso de canales virtuales. Como en los anteriores, es un algoritmo de encaminamiento Path-like, es decir, para alcanzar el conjunto de destinos, el mensaje multicast no se bifurca en los nodos intermedios.

A diferencia de los anteriores, la función de partición y ordenación de destinos (preparación del mensaje) sólo realiza una ordenación, no divide el conjunto de destinos en subconjuntos. Esta ordenación de los destinos es muy importante para asegurar que el algoritmo es completamente libre de bloqueos. Se realiza atendiendo a las siguientes especificaciones:

Partiendo del conjunto de destinos, se escoge el destino con coordenada *x* menor, d_{xmin}. Si este destino tiene su coordenada *x* menor que la del origen, entonces se encamina hacia él por ruta determinista usando encaminamiento *XY* (si en esta ruta existen destinos para este mensaje, el mensaje entraría también en ellos). Una vez alcanzado d_{xmin}, se encamina el mensaje hacia el resto usando encaminamiento adaptativo de ruta mínima, de forma que el mensaje viaje en dirección Este (nunca en dirección Oeste). Si dos destinos tiene la misma *x*, entonces se les visita de Norte a Sur.

El algoritmo asociado podría ser el siguiente:

Entrada: Conjunto de destinos D, $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$ y un nodo origen $u_0 = (x_0, y_0)$.

Salida: Lista ordenada de destinos, M_H.

Procedimiento:

1. $K = D\ ; (x_h, y_h) = (x_0, y_0); i = 0;$
2. Mientras $K \neq 0$ hacer:
   (a) Si $\exists\ (x,y) \in K\ /\ x < x_h \wedge y = y_h$ entonces seleccionar $(x_j, y_j)\ /\ x_j = \max\{x_l\ /\ (x_l, y_l) \in K,\ x_l < x_h,\ y_l = y_h\}$ e ir a (d).
   (b) Si $\exists\ (x,y) \in K\ /\ x = \min\{x_l / (x_l, y_l) \in K\} \wedge y \geq y_h$ entonces seleccionar $(x_j, y_j)\ /\ x_j = \min\{x_l\ /\ (x_l, y_l) \in K\} \wedge y_j = \min\{y_m\ /\ (x_m, y_m) \in K,\ x_m = x_j,\ y_m \geq y_h\}$ e ir a (d)
   (c) Si $\exists\ (x,y) \in K\ /\ x = \min\{x_l\ /\ (x_l, y_l) \in K\} \wedge y < y_h$ entonces seleccionar $(x_j, y_j)\ /\ x_j = \min\{x_l\ /\ (x_l, y_l) \in K\} \wedge y_j = \max\{y_m\ /\ (x_m, y_m) \in K,\ x_m = x_j,\ y_m < y_h\}$
   (d) $M_H[i] = (x_j, y_j);\ i = i + 1;\ K = K - \{(x_j, y_j)\};\ (x_h, y_h) = (x_j, y_j);$
3. Colocar M_H en la cabecera del mensaje.

El algoritmo de encaminamiento se adapta a esta forma de ordenar los destinos, para ello debe ser determinista para alcanzar a d_{xmin} (y todos los que se encuentre en este camino), y después adaptativo para alcanzar al resto en dirección Este-Oeste. El algoritmo podría ser el siguiente:

$R : NxN \rightarrow N$, se define como $R(u, v) = w$ de forma que:
Si $x_v \geq x_u$, entonces w es cualquier vecino en la ruta mínima hacia v
Si $x_v < x_u$, entonces $w = (x_u - 1, y_u)$

En la *figura 8* tenemos un ejemplo del encaminamiento PM para una malla 2D, donde se envía un mensaje multicast con 5 destinos desde el nodo (2,1).

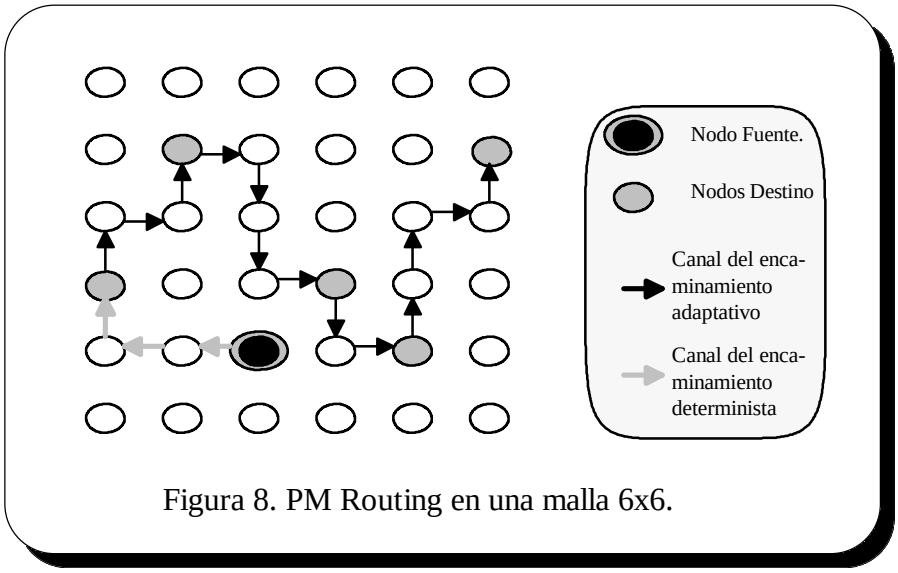

Figura 8. PM Routing en una malla 6x6.

Al usar un único mensaje multicast para todos los destinos, es posible que la ruta hasta el último sea muy larga. Se propone una variación del encaminamiento PM donde se realiza una partición del conjunto de destinos original (como en Dual-Path): un subconjunto para los destinos al Oeste del origen y otro para los destinos al Este del mismo.

5.2 Algoritmo FM.

Este algoritmo es similar al anterior pero suministra una adaptatividad total. Para ello no queda mas remedio que añadir canales a la red (normalmente multiplexando canales físicos: canales virtuales). En este caso se duplican solamente los canales verticales de la malla, obteniendo por cada canal físico vertical dos canales que etiquetamos como v_1 y v_2. Con esto se divide la red en dos conjuntos de canales:

$C_W = \{c \,/\, c$ es un canal horizontal de Este a Oeste ó $c \in C_{v_1}\}$
$C_E = \{c \,/\, c$ es un canal horizontal de Oeste a Este ó $c \in C_{v_2}\}$
donde
C_{v_1} y C_{v_2} son, respectivamente, conjuntos de canales verticales tipo v_1 y v_2 .

Básicamente se divide el conjunto de destinos en dos subconjuntos M_{HW} y M_{HE} , de forma que M_{HW} contiene los destinos que se encuentran al oeste del origen y que se alcanzan usando canales de $C_{w.}$ (de manera análoga se construye M_{HE}). El algoritmo de encaminamiento es equivalente al que vimos para PM.

En la *figura 9* tenemos el mismo ejemplo que habíamos propuesto para PM pero con este nuevo encaminamiento.

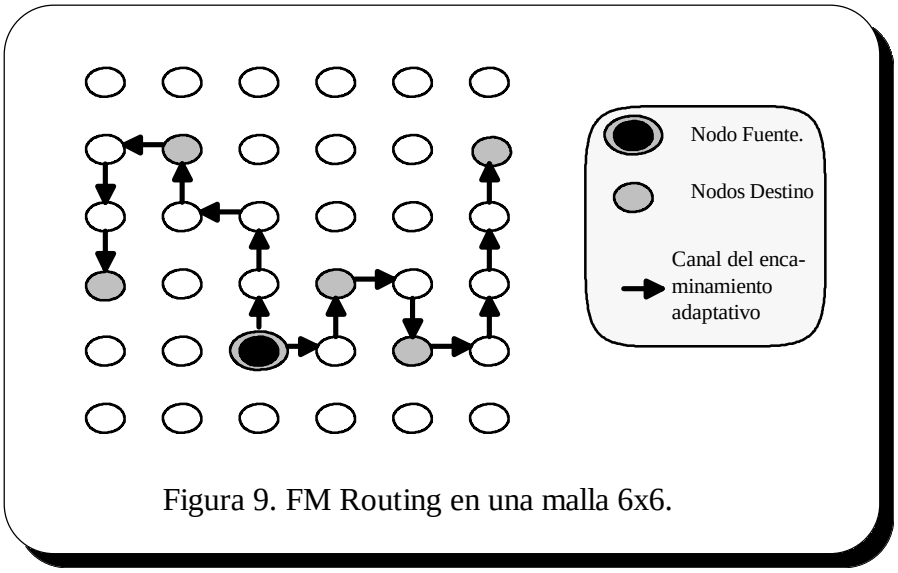

Figura 9. FM Routing en una malla 6x6.

5.3 Teoría para el diseño de algoritmos multicast adaptativos y libres de bloqueos.

Como hemos visto, no es obvio diseñar buenos algoritmos multicast adaptativos que además sean libres de bloqueos. Para facilitar esta tarea, *J. Duato* [8], desarrolló una teoría para el diseño de algoritmos multicast adaptativos y libres de bloqueos. En ella, se analizan las nuevas dependencias de canales que surgen al introducir multicast, y se intenta definir un nuevo grafo de dependencias multicast y su correspondiente extendido (de forma análoga a [9]). Se dice que:

> *Si el grafo de dependencias multicast extendido de una red, con un algoritmo de encaminamiento y una función de partición de destinos, no tiene ciclos, el algoritmo de encaminamiento es libre de bloqueos.*

Hay que señalar que, tanto el algoritmo de encaminamiento como el de partición contribuyen a la aparición de dependencias multicast que tenemos que considerar al construir el nuevo grafo de dependencias multidestino. Por tanto, los ciclos del grafo de dependencias se pueden eliminar imponiendo restricciones sobre el algoritmo de encaminamiento y/o la función de partición de destinos.

En la *figura 10*, se muestra un ejemplo de un algoritmo que sigue esta metodología basándose en un algoritmo determinista del tipo Dual-Path.

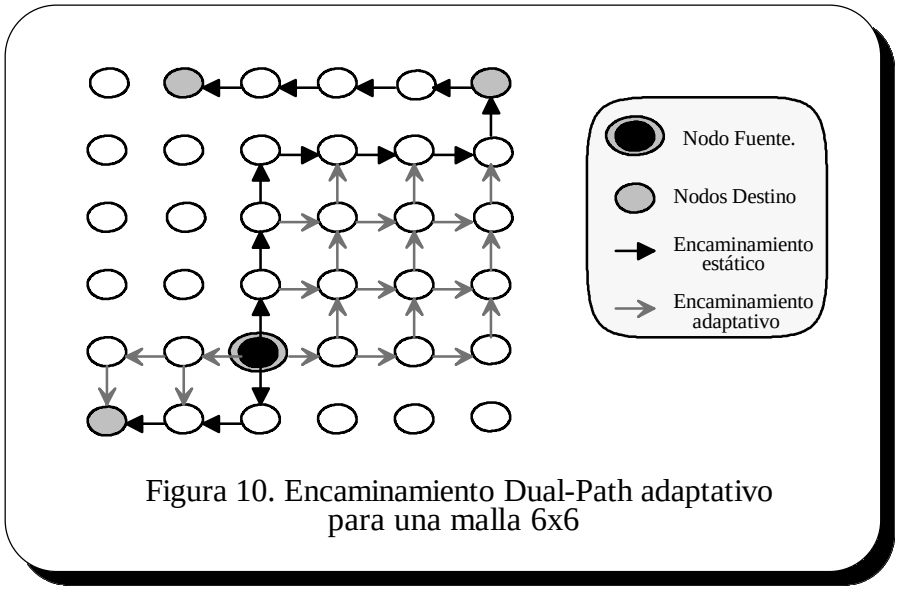

Figura 10. Encaminamiento Dual-Path adaptativo para una malla 6x6

La idea para conseguir un algoritmo multidestino libre de bloqueos podría ser la siguiente: Nos basamos en un algoritmo determinista que sea libre de bloqueos, y añadimos nuevos canales a la red (canales virtuales), de forma que un canal se reserva para la versión determinista y el resto se usan de forma adaptativa. Así, si evitamos que aparezcan ciclos de dependencias de canales dentro del conjunto de canales de escape (reservados para el determinista), aunque existan bucles en el grafo de dependencias de canales extendido, siempre existirá una ruta de escape que pueda deshacer el posible bloqueo.

En el ejemplo (*figura 10*), utilizamos un algoritmo del tipo Dual-Path ya conocido y duplicamos cada canal físico de la red, apareciendo canales A y B de cada canal físico original. Los canales A los usamos siguiendo el encaminamiento Dual-Path, y el resto siguiendo un encaminamiento adaptativo de ruta mínima (con una excepción, en la ruta al destino no podemos usar nodos intermedios con una etiqueta mayor).

6. Simulador multicast.

Para hacer un estudio completo de los algoritmos de encaminamiento multicast y de las funciones de partición de destinos en diferentes topologías y bajo diversas condiciones, es necesario disponer de una herramienta que nos proporcione resultados válidos sobre el comportamiento de los diferentes elementos que intervienen en una red de interconexión.

Por ello, como primer paso, se ha desarrollado un simulador de redes de interconexión que contemple las características particulares de las comunicaciones Unicast, Multicast y Broadcast en una red de interconexión con control de flujo Wormhole Routing.

Este simulador es una extensión del ya desarrollado por *J.Duato*, que incluye los mecanismos de difusión de mensajes que hemos descrito anteriormente.

6.1. Descripción del simulador.

La generación de mensajes es uniforme y configurable. Un nodo, en un instante dado, generará un mensaje Unicast/Multicast que podrá ser encaminado por el router si se dispone de canales internos de memoria libres. A la hora de generar un mensaje en un nodo se siguen los siguientes pasos:

1.- Se calcula el número de destinos a los que va dirigido el mensaje.

2.- Se generan el conjunto de destinos D de forma aleatoria.

3.- Se efectúa la partición de destinos que nos indicará el número de mensajes que van a partir de ese nodo.

4.- Se comprueba si tenemos canales internos de memoria libres, para enviar al router los mensajes generados en 3.

Cada mensaje tiene una cabecera que consta de uno (Unicast) o más flits (Multicast), como muestra la *figura 11*.

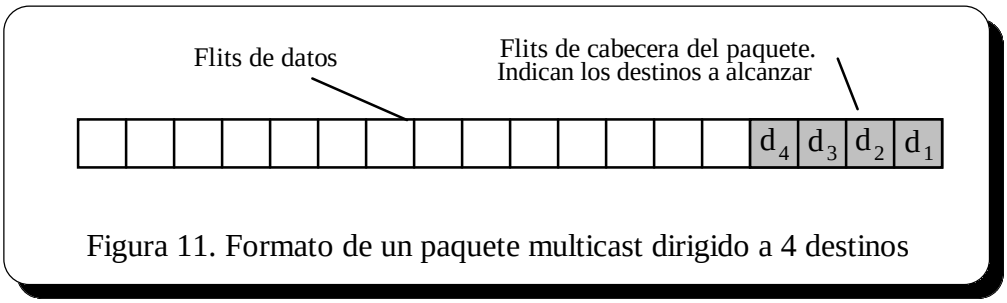

Figura 11. Formato de un paquete multicast dirigido a 4 destinos

Cuando la cabecera de un mensaje entre en un nodo, el router de ese nodo intentará encaminarla. Si se dispone del siguiente canal de salida, seleccionado por el algoritmo de encaminamiento, entonces se pasa al siguiente nodo. En caso contrario, el mensaje asociado a esa cabecera se bloquea en el nodo hasta que se libere el canal deseado.

Si al encaminar la cabecera del mensaje el canal seleccionado es interno, de entrada a la memoria del nodo, y además está disponible, entonces:

1.- Se elimina el primer flit de la cabecera.

2.- Si quedan destinos pendientes (más flits de la cabecera) entonces esperamos a que llegue la segunda cabecera, sino el mensaje habrá llegado al último destino.

3.- Cuando tengamos la siguiente cabecera, intentaremos encaminarla para alcanzar el siguiente destino.

4.- Si se dispone del canal de salida requerido se encamina hacia el siguiente destino, sino se bloqueará hasta que el canal se libere.

Según va avanzando la cabecera del mensaje multicast por la red, los flits restantes la siguen por la ruta que va abriendo. Cuando la última cabecera llega al destino, la ruta estará establecida. Los canales asignados se liberarán cuando el último flit del mensaje los atraviesen.

Cuando un nodo intermedio de la ruta de un mensaje multicast pertenece al conjunto de destinos del mismo, se produce una bifurcación en la ruta. Es decir, cada flit que entre al router debe ser copiado sobre el canal interno de entrada a memoria y sobre el canal externo de salida al siguiente nodo de la ruta (ver *figura 12*).

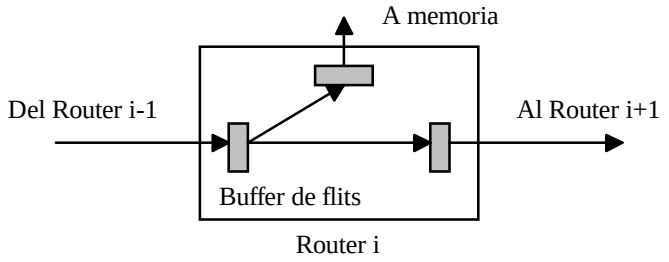

Figura 12. Bifurcación en un destino intermedio.

Se han incluido los algoritmos de encaminamiento así como las funciones de partición de destinos de *Dual-Path*, *Multi-Path* y *PM* con el objeto de realizar varias simulaciones y verificar su comportamiento en mallas 2D.

6.2. Análisis de resultados de simulación.

Para la realización de las simulaciones se han considerado los siguientes parámetros:

- Malla 2D de 1024 nodos
- Tamaño de los mensajes = 16 flits.
- Número de destinos/mensaje = 16.
- Número de canales de memoria/nudo = 4
- Tiempos de encaminamiento, transimisión y cruce por el crossbar unitarios (= 1 ciclo de reloj).
- La generación de mensajes es uniforme.

En la *figura 13* se muestra las curvas comparativas de varios algoritmos multidestino. En concreto se analizan los algoritmos:

- Dual-Path determinista.
- Dual-Path adaptativo (con dos canales virtuales).
- Dual-Path estático (con dos canales virtuales).
- PM (parcialmente adaptativo).

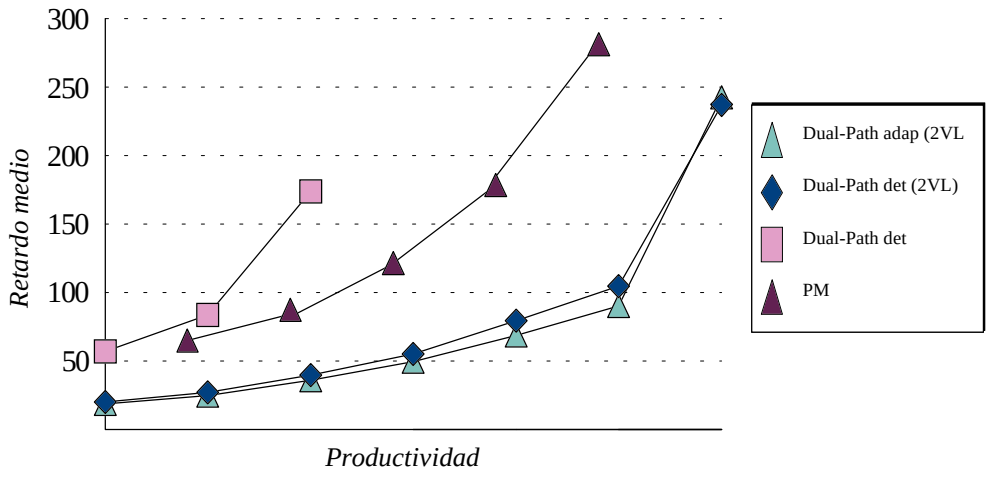

Figura 13: Simulaciónes multicast sobre una malla2D (1Knode)

De las gráficas podemos deducir que añadiendo adaptatividad al Dual-Path en una malla 2D no se consigue mejorar apreciablemente las prestaciones del mismo. En la comparación se tiene en cuenta el algoritmo Dual-Path con dos canales virtuales, ya que sólo queremos ver la mejora que introduce sólo la adaptatividad del correspondiente algoritmo adaptativo, construido según la metodología descrita en [8].

Se aprecia que Dual-Path determinista se satura rápidamente en cuanto aumentamos la carga de la red. Por último, el algoritmo PM tiene grandes retardos en los mensajes ya que la ruta desde el origen hasta el último destino puede ser excesivamente larga (no hay partición de mensaje, sólo la ordenación correspondiente). En comparación con el puramente determinista aguanta mucho mejor el aumento del tráfico.

De cara a una implementación hardware de estos algoritmos, estos deben soportar también la comunicación Unicast (uno-a-uno) y así con un único algoritmo de encaminamiento se contemplan todas las formas de comunicación. Por esta razón, tendremos que comprobar el rendimiento de estos algoritmos con mensajes Unicast, que ,en realidad, serán la mayoría de mensajes que circulen por la red (se estima que puede ser sobre el 90% de los mensajes).

En la *figura 14*, se muestra una comparativa entre los algoritmos:

- Dual-Path determinista.
- Dual-Path adaptativo (con dos canales virtuales).
- XY determinista.
- XY adaptativo (con dos canales virtuales).
- PM (parcialmente adaptativo).

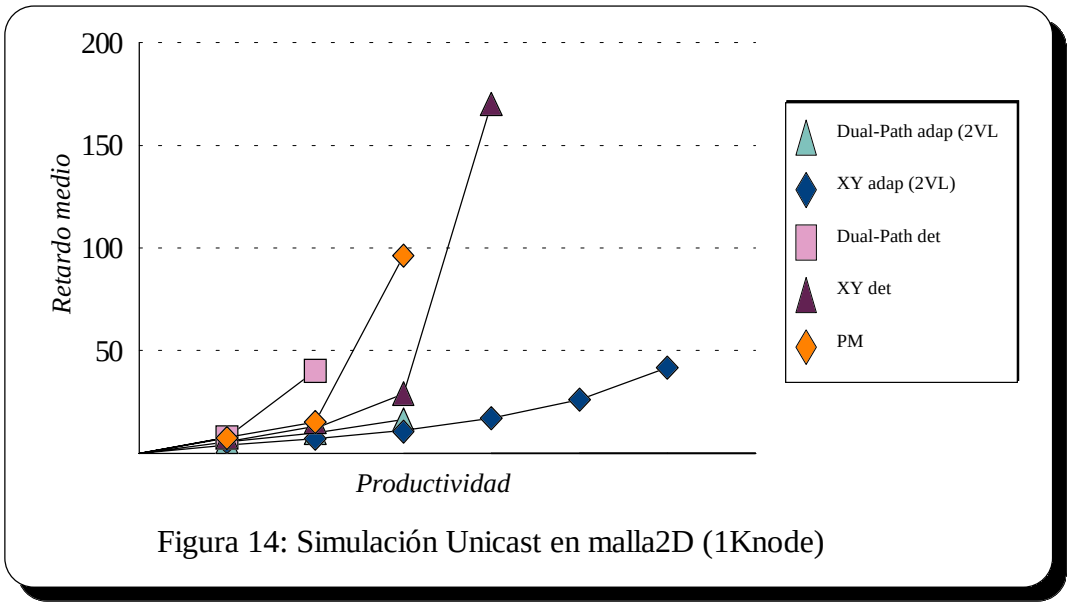

Figura 14: Simulación Unicast en malla2D (1Knode)

Se puede ver que Dual-Path adaptativo llega a servir la mitad de tráfico que el XY adaptativo, saturándose rápidamente. Sin embargo hasta la saturación tiene un comportamiento similar al XY adaptativo.

Por su parte, el algoritmo PM se comporta algo peor que el Dual-Path adaptativo y el Dual-Path determinista.

En definitiva, lo algoritmos que se han considerado pueden trabajar sobre multicast pero su versión Unicast no es muy buena. Por otro lado, en multicast el introducir mayor adaptatividad en los algoritmos no incrementa mucho las prestaciones de los mismos. Todo esto se debe aplicar al uso de mallas 2D. Si utilizásemos otras topologías con mayor grado de conectividad quizás estos algoritmos se comportasen mejor que con las mallas.

7. Conclusiones.

Se han estudiado las técnicas básicas para la generación de mensajes multidestino, comenzando por la forma de realizar la difusión: *Tree-like* o *Path-like routing*. Se ha visto que es más conveniente utilizar éste último de cara a una mejor utilización de los recursos de la red y una implementación más sencilla. Hemos ido conociendo algunos algoritmos de encaminamiento y partición-ordenación de destinos, observando que es tan importante la función de encaminamiento como la de partición-ordenación a la hora de diseñar algoritmos libres de bloqueos.

Para formalizar el diseño de algoritmos multidestino libres de bloqueos y hacerlos más flexibles y efectivos se elabora una teoría que permite desarrollar dichos algoritmos garantizando la ausencia de bloqueos. Se comprueba que los algoritmos que se realizan a partir de esta son mejores que los tradicionales en todos los sentidos, aunque en las mallas 2D no obtengan todas las ventajas que podrían obtener en redes con mayor conectividad.

Por último, se habla de una herramienta de simulación con las extensiones multidestino, que hemos utilizado para comparar el rendimiento de varios de los algoritmos propuestos sobre una malla 2D tanto para mensajes Multicast como para mensajes Unicast.

8. Referencias bibliográficas.

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

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

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

[4] 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.

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

[6] 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.

[7] 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.

[8] J. Duato, "*On the design of deadlock-free adaptive multicast routing algorithms*". Proc. Parallel Arquitectures Languages Europe, Jun. 93.

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