ESTUDIO DE LA RECONFIGURACIÓN DINÁMICA DE LA TOPOLOGÍA DE INTERCONEXIÓN EN UNA RED DE TRANSPUTERS

José M. García y José Duato

Resumen

Los multicomputadores y, en particular, las redes de transputer son un tipo de máquinas paralelas cada dia más populares, debido a que poseen una excelente relación prestaciones/coste. Su principal problema radica en la red de interconexión cuando el número de comunicaciones entre los nodos es elevado. En este artículo presentamos una nueva forma para resolver este problema: la reconfiguración dinámica de la topología de la red de interconexión. Para comprender perfectamente estas ideas describimos el funcionamiento de un algoritmo que nosotros hemos desarrollado. Esta técnica es además muy adecuada para aplicaciones paralelas cuyo patrón de comunicaciones varía a lo largo del tiempo.

Abstract

Multicomputers and transputer networks are becoming very popular, because they have a very good performance/cost ratio. The interconnection network is the bottleneck for these machines, especially when processes exchange messages very frequently. In this paper we present a new way to solve this problem: the dynamic reconfiguration of the topology of the interconnection network. To understand how it works we also describe a reconfiguration algorithm we have developed. Moreover, this new approach is very well suited for parallel applications whose communication pattern varies over time.

1. INTRODUCCIÓN.

Los multicomputadores [2] son computadores con varios procesadores, cada uno con su propia memoria local, que se comunican entre sí a través de una red de interconexión.

Estas máquinas tienen poco más de una década de existencia, habiéndose desarrollado el primer prototipo (Cosmic Cube) en 1981 [19]. A

pesar de su corta historia, han sufrido un desarrollo muy rápido, distinguiéndose ya dos generaciones.

Su tardía aparición cabe atribuirla a tres factores fundamentales: los requisitos de memoria, la comunicación entre procesadores y la dificultad de programación. Contrariamente a los multiprocesadores con memoria compartida, el código a ejecutar por cada procesador (sistema operativo y procesos de usuario) debe estar replicado en todas las memorias locales. Esto implica una cota inferior para el tamaño de memoria en cada procesador. Por ello, no se emprendió la construcción de estas máquinas hasta la aparición en el mercado de memorias RAM de bastante capacidad a un precio asequible. El primer prototipo tenía 128Kb por procesador, mientras que los multicomputadores actuales suelen tener de 4Mb a 64Mb por procesador.

Por contra, los multiprocesadores (máquinas paralelas con memoria compartida) tuvieron inicialmente un desarrollo más rápido. Sin embargo, tienen la desventaja frente a los multicomputadores de su poca escalabilidad. El acceso a una memoria compartida por parte de varios procesadores requiere complejas organizaciones de memoria, lo cual supone un coste muy elevado y limita en la práctica el número de procesadores a unas pocas decenas. En cambio, existen multicomputadores comerciales con miles de procesadores.

En los multicomputadores, la respuesta que tenga la red de interconexión es un factor determinante en la potencia y velocidad de respuesta de estas máquinas. Ello obliga a un eficiente manejo de las comunicaciones entre los procesadores, de cara a aumentar las prestaciones de las máquinas. Por tanto, trabajan bien con problemas cuyo ratio cálculo/comunicaciones (por nodo) sea alto.

Un aspecto decisivo en el progreso de los multicomputadores es la mejora de las técnicas de encaminamiento de los mensajes a través de la red de interconexión. Los progresos en los algoritmos de encaminamiento mediante la utilización de algoritmos adaptativos y tolerantes a fallos [9], y sobre todo, la mejora en las técnicas del control del flujo de mensajes en la red, han sido hechos decisivos que han aumentado las prestaciones de la red de interconexión. Tanto es así, que el paso de una a otra generación en los multicomputadores se define en función de la técnica de control de flujo de mensajes que usan [2]: un mecanismo llamado de *almacenamiento y reenvío*[1] en el caso de la primera generación y un mecanismo denominado *encaminamiento segmentado con espera*[2] [6] en el caso de la segunda generación. Este mecanismo de control de flujo tiene dos ventajas importantes respecto al mecanismo de almacenamiento y reenvío: no requiere el uso de la memoria local en los nudos intermedios y la latencia de los mensajes se ve poco afectada por la distancia a recorrer. Estas características se consiguen debido a que en este mecanismo cada mensaje se descompone en pequeños fragmentos denominados unidades de control de flujo o *flits* [5], segmentando el envio de los mismos.

En la actualidad, los computadores masivamente paralelos con miles de procesadores son considerados como la tecnología más prometedora para alcanzar una capacidad de procesamiento cercana a los teraflops. Incluso en el caso de que dichos procesadores se organicen en una arquitectura con memoria compartida, el principal cuello de botella de estas máquinas es la red

---

[1]En inglés "store-and-forward"

[2]En inglés "wormhole routing"

de interconexión. Por ello, la mejora en el diseño e implementación de la red de interconexión es cada vez más importante en una arquitectura paralela, ya sea con memoria compartida o distribuida.

En este artículo pretendemos ofrecer un enfoque alternativo a la solución de mejorar las prestaciones de la red. Además de todas las mejoras tecnológicas que se apliquen en la red, una alternativa para mejorar su rendimiento es disminuir el número de mensajes que circulan a través de ella. Una forma de llevar esto a cabo es mediante la reconfiguración dinámica de la topología de la red de interconexión. La idea central de este enfoque es procurar que los nodos que tienen una tasa de comunicaciones elevada estén próximos en la red. Para ello, dinámicamente -es decir, a lo largo de la ejecución de una aplicación y en función de las necesidades que ésta presente- se va reconfigurando la topología para conseguir este objetivo. Con esta nueva técnica lo que pretendemos es *acercar* los nodos que más comunican entre sí para disminuir el número de mensajes que circulan por la red. Este enfoque es muy adecuado para problemas paralelos cuyo patrón de comunicaciones varía a lo largo del tiempo. Estos son frecuentes en diversas aplicaciones de cálculo numérico y matricial, así como en aplicaciones relacionadas con la visión, donde se presentan diversas fases con unos requerimientos de comunicaciones distintos de unas fases a otras.

En este artículo vamos a describir cómo se puede implementar esta novedosa técnica teniendo como punto de referencia una máquina real basada en transputers, el PARSYS SN 1000. En el apartado 2 expondremos con mayor profundidad el problema de las comunicaciones y las formas habituales de solucionarlo para, en el siguiente apartado, presentar la técnica de la reconfiguración dinámica. Asimismo, se presentará el protocolo de

reconfiguración que nosotros hemos desarrollado y cómo se puede implementar en el SN 1000. En el apartado 4 se detallará el funcionamiento del algoritmo de reconfiguración por medio de un sencillo ejemplo y de un algoritmo paralelo más complejo, mostrando los buenos resultados que se obtienen. Antes de terminar el artículo, presentaremos cuál es el estado del arte de este tema, y por último finalizaremos con un apartado de conclusiones.

2. EL PROBLEMA DE LAS COMUNICACIONES.

Como ya se ha descrito en la introducción, los multicomputadores son máquinas adecuadas para trabajar con problemas cuyo ratio cálculo/comunicaciones sea alto, es decir, con una tasa baja de comunicaciones entre nodos.

El problema fundamental de las comunicaciones es el tiempo de retraso que introducen dos procesos cada vez que comunican, y que afecta a la potencia total del multicomputador. Por lo tanto, y de cara a reducir el tiempo total de ejecución de un algoritmo, se ha estudiado el modo de disminuir el tráfico de mensajes a través de la red de interconexión. Este problema se ha estudiado tradicionalmente desde tres puntos de vista diferentes: la topología de la red de interconexión, el algoritmo de encaminamiento de mensajes a través de la red y la asignación de los procesos sobre los procesadores físicos (mapeo). Con una topología adecuada se puede reducir la distancia entre los nodos que comunican entre sí. Debido a problemas tecnológicos y de coste, no es posible tener la topología ideal (totalmente conectada). Para una topología dada, se puede reducir el tiempo de comunicación con un buen algoritmo de encaminamiento. Por último, para una topología concreta y un algoritmo de encaminamiento dado, las comunicaciones se reducen si los procesos se

asignan a los procesadores físicos adecuadamente, es decir, los procesos que comunican con mayor frecuencia se asignan a procesadores que estén físicamente conectados.

Un compendio de las topologías clásicas que se han adoptado en los multicomputadores, estudiando los pros y los contras de cada una, se puede encontrar en [10]. Por otra parte, el diseño de algoritmos de encaminamiento es bastante delicado, ya que debe garantizarse que no puedan aparecer situaciones de bloqueo. En la actualidad se está trabajando en algoritmos adaptativos, que ofrecen mejores prestaciones en los multicomputa-dores para aplicaciones con patrón de comunicaciones no uniforme. Nosotros hemos desarrollado teorías que permiten garantizar la ausencia de bloqueo en algoritmos adaptativos con dependencias cíclicas entre canales, lo cual permite una mayor flexibilidad y mayores prestaciones [8,9].

El problema del mapeo (asignación) de procesos a procesadores fué tratado por primera vez de forma extensa en [4]. Para problemas cuyo patrón de comunicaciones varía a lo largo del tiempo se emplea la asignación dinámica. Un estudio interesante sobre este punto se pueden encontrar en [17]. Uno de los principales problemas que tiene la asignación dinámica de procesos es que se produce una *migración* de procesos a través de la red, donde no sólo se debe mover el código correspondiente a ese proceso sino que también se deben mover los valores de las variables de ese proceso, dado que los procesadores no tienen memoria compartida.

Una alternativa a estas tres soluciones que acabamos de describir para mejorar la comunicación entre procesos, es la **reconfiguración dinámica de la red de interconexión**. Para aquellos problemas cuyo patrón de comunicaciones no es regular, en vez de *migrar* procesos de unos

procesadores a otros, se puede modificar la topología de la red para que se adapte al patrón de comunicaciones que haya en ese momento.

Bajo este criterio, los multicomputadores pueden ser clasificados como sigue:

* *Topología estática*: Las conexiones entre nodos se realizan por enlaces fijos punto-a-punto, no modificables. Ejemplos de esta clase de sistemas son sencillos sistemas con transputers o la mayoría de las máquinas con topología hipercubo o malla [2].

* *Topología cuasi-estática*: Una red con conmutadores (*switching network*) permite establecer una topología específica para un programa determinado *antes* de que comience su ejecución. Durante la ejecución del programa la topología ya no es modificable. En [21] se describe un trabajo de este tipo basado en un Supernode con 16 transputers T414.

* *Topología dinámica*: La topología de la red puede cambiar casi arbitrariamente durante la ejecución de los programas, pudiendo fácilmente ajustarse al modelo de comunicaciones de un programa determinado. Otra ventaja es la tolerancia a fallos: los nodos o enlaces que hayan fallado pueden ser fácilmente arrinconados realizando un *by pass* por ellos. Ultimamente, también se está estudiando un sistema de transputers reconfigurable con múltiples niveles. En [20] se describe un sistema de este tipo denominado DIRECT.

Básicamente, la principal diferencia entre la topología *dinámica* y la *cuasi-estática* es el tiempo necesario para reconfigurar la red. Por otro lado, las topologías dinámicas usualmente ofrecen la posibilidad de realizar una reconfiguración parcial de la red, siendo esta operación más rápida que la reconfiguración global. Debido a que la reconfiguración dinámica de la red

conlleva un coste, no es posible reconfigurar la red cada vez que se envíe un mensaje [3] ya que entonces no obtendríamos ninguna mejora. El problema reside entonces en el modo de establecer cuando un par de procesadores van a intercambiar una cantidad considerable de mensajes. Hay dos formas de resolver esto:

- El programador inserta unas primitivas de cambio de topología en aquellos puntos en que conoce que es adecuado realizar un cambio. Este enfoque es adecuado para problemas cuyo patrón de comunicaciones varía con el tiempo pero *a priori* ya se sabe cuál es la topología más conveniente en cada una de las fases. A veces también recibe el nombre de topología *cuasi*-dinámica.

- Por medio de un algoritmo de reconfiguración que sea el responsable de decidir cuando es más adecuado modificar la topología, en base a alguna función de coste. Esta segunda forma se le denomina en ocasiones topología *verdaderamente* dinámica. Este segundo caso abarca un mayor tipo de problemas y es donde nosotros hemos centrado nuestros estudios.

3. DESCRIPCIÓN DEL PROTOCOLO DE RECONFIGURACIÓN.

En este apartado vamos a describir cómo se puede llevar a cabo la reconfiguración dinámica de la red. Para ello, vamos a describir el protocolo que hemos desarrollado para el caso de una máquina comercial. Un estudio más extenso de dicho protocolo se puede encontrar en [11].

Esta línea de investigación tuvo su origen en el intento por mejorar el rendimiento de las comunicaciones en una máquina basada en transputers, el PARSYS SN 1000. Debido a ello, el algoritmo desarrollado para controlar la reconfiguración dinámica está influenciado por las características de este

multicomputador. Por ejemplo, en la reconfiguración de la red sólo se permite que los procesadores intercambien sus posiciones con los nodos vecinos, pues esta es una de las facilidades que presenta el Supernodo.

La idea básica del algoritmo es la siguiente: cuando el tráfico entre un par de nodos es intenso, el algoritmo intentará colocar el nodo origen cerca del nodo destino mediante el intercambio de las posiciones en la red de, o bien el nodo origen y uno de sus vecinos, o bien el nodo destino y uno de sus vecinos. Sin embargo, la implementación de este algoritmo es distribuida, teniendo en cuenta cada nodo únicamente la información que él tiene registrada localmente.

Antes de pasar a describir dicho protocolo, vamos a comentar brevemente algunas características de la arquitectura del SN1000 en las que se basa.

3.1 La arquitectura del supernodo.

El Supernodo es un multicomputador MIMD reconfigurable basado en transputers, que ha sido el resultado del proyecto ESPRIT P1085 [16]. Esta máquina puede tener de 16 a 1024 transputers. La configuración básica consta de 16 (32) transputers (T800), cada uno de ellos con su memoria local, y un transputer de control (T414). Los transputers comunican entre ellos a través de un *switch* programable que permite adoptar diversas topologías para la red de interconexión. Los transputers y el controlador también están conectados a través de un **bus de control**. Configuraciones más grandes pueden ser generadas recursivamente por combinación de estas formas básicas.

El *switch* programable es funcionalmente equivalente a un par de *switches* elementales de tamaño 32x32. Si se denominan a los cuatro enlaces

de los transputers *norte, sur, este y oeste,* el primer *switch* elemental controla las conexiones norte/sur, mientras que el segundo trata con las conexiones este/oeste. En [16] se muestra que esta arquitectura es lo bastante flexible para permitir que cualquier topología de grado cuatro pueda ser implementada. La programación del *switch* sólo es posible desde el controlador (en la literatura se le denomina también *system controller*). En la figura 1 se puede ver un esquema del switch.

Los transputers se conectan al controlador a través del bus de control. En [1] se sugiere que este bus también puede ser usado para mejorar la efectividad de todo el sistema (sincronización entre transputers) o para la depuración. En nuestro caso, y como se detallará a continuación, va a jugar un papel fundamental de cara a la reconfiguración dinámica del *switch*. En la figura 2 se puede ver un esquema del bus de control.

3.2 El protocolo de reconfiguración.

A continuación se va a presentar el protocolo que nosotros hemos desarrollado para manejar la reconfiguración dinámica de la red de interconexión. Dicho protocolo tiene dos partes diferentes: una que se ejecuta en el *system controller* y otra que se ejecuta en el resto de nodos del multicomputador. Estos algoritmos forman parte del núcleo del *run-time*, ya sea en el *system controller* o en el resto de nodos del sistema.

El algoritmo que se va a ejecutar en los nodos solo se activará en aquellos nodos que reciban un mensaje; es decir, cuando un nodo envía un mensaje sólo registra dicha información sin ejecutar el algoritmo de reconfiguración, y es cuando recibe un mensaje cuando está en condiciones

de ejecutar dicho algoritmo. Si tras la ejecución del algoritmo se superan unos umbrales mínimos que tiene, y además la función de coste resulta positiva, se iniciará un proceso de reconfiguración cuyo fin será acercarlo al nodo emisor de mensajes.

En este modelo, el nodo responsable de la reconfiguración (el *system controller*) realiza el control del cambio de topología por medio del bus de control.

El protocolo de reconfiguración de la red trabaja del siguiente modo:

1. Cuando un nodo decide que es necesario reconfigurar la red (lo cual se determina mediante dos umbrales y una función de coste [11]), envía una señal al *system controller* a través del bus de control.

2. A continuación, el *system controller* comunica a todos los nodos que se va a producir una reconfiguración y que por tanto dejen de comunicar mensajes entre ellos. Para minimizar el tiempo de reconfiguración y aliviar el coste que supone, se dejan de transmitir incluso los mensajes que están en nodos intermedios. Una vez que se haya reconfigurado la red, dichos mensajes se encaminarán a su nodo destino.

3. El nodo que ha pedido la reconfiguración envía al *system controller* los datos de la reconfiguración para que se pueda realizar la modificación de la topología de la red.

4. Con estos datos, y una vez todos los nodos le han indicado que han dejado de enviar mensajes, el *system controller* modifica la topología de la red de interconexión adaptándola a las nuevas circunstancias.

5. Una vez que la nueva configuración ha sido establecida, el *system controller* envía dicha configuración a todos los nodos (para que puedan encaminar correctamente los mensajes) y permite de nuevo las

comunicaciones entre los nodos.

Este protocolo se puede implementar fácilmente usando el bus de control disponible en la arquitectura del Supernodo, lo cual no añade tráfico de mensajes a la red cuando se reconfigura la topología. Más aún, el uso del bus de control permite realizar eficientemente las operaciones de difusión de información requeridas en los pasos 2 y 5.

Una puntualización acerca del modo de controlar la reconfiguración. El control centralizado podría ser un cuello de botella para todo el sistema, ya que todos los nodos deben de pedir al *system controller* que supervise los cambios. Esta situación sería verdad en el caso de que se reconfigurara la red con mucha frecuencia. En el caso de sistemas moderados (por ejemplo hasta 64 nodos) las peticiones de cambio no van a ser excesivas, y por tanto el control centralizado no va a representar ningún cuello de botella para el sistema. Para sistemas más grandes, la arquitectura del Supernodo necesitaría un *switch* de dos niveles y más de un bus de control.

Hasta aquí la somera descripción del protocolo para llevar a cabo la reconfiguración de la red. Dicho protocolo tiene una serie de propiedades acerca del modo en que ejecuta la reconfiguración. Dichas propiedades son las siguientes: reconfiguración local, preserva la topología de la red, se basa en una función de coste, produce una pequeña alteración en la red, usa dos umbrales para determinar la reconfiguración de la red y gobierna la reconfiguración mediante un control centralizado. La función de coste es la que se encarga de evaluar el tráfico en la red. Para ello, tiene en cuenta dos aspectos importantes: el *peso* de la comunicación y la *distancia* entre los nodos. Por el *peso* de la comunicación se entiende el número de mensajes que

se han mantenido en una determinada dirección. La *distancia* entre nodos mide el número de nodos intermedios que un mensaje debe atravesar para ir de un nodo A a un nodo B; si A y B están físicamente conectados la distancia es nula. Entonces, la función de coste para un nodo cualquiera es la suma del peso de las comunicaciones que mantiene con el resto de nodos ponderadas por la distancia relativa que hay a esos nodos. Un estudio de las diversas propiedades que posee nuestro algoritmo se pueden encontrar en [11].

A la hora de evaluar el rendimiento que se obtiene con la reconfiguración de la red, es necesario conocer (o estimar) cuánto es el tiempo perdido cada vez que se realiza un cambio en la topología de la red. Este aspecto lo hemos estudiado en [12], donde se evalúa una cota superior para el tiempo que necesita la máquina para realizar un cambio en la topología de la red. La evaluación de los tiempos se ha realizado considerando como máquina objeto el Supernodo. Los resultados obtenidos son muy satisfactorios tanto para algoritmos triviales como para complejos algoritmos numéricos que se han desarrollado.

Las posibilidades de manejar la reconfiguración de la red son múltiples y variadas. En [14] hemos realizado un análisis de las diferentes opciones que tenemos, justificando el por qué de las soluciones adoptadas. Asimismo, hemos estudiado los problemas que presenta esta técnica y los posibles límites que se pueden encontrar.

4. UN EJEMPLO PRÁCTICO.

En este apartado pretendemos completar la exposición del funcionamiento de la reconfiguración de la red por medio de un ejemplo. Gracias a dicho ejemplo ilustraremos cómo se comporta el algoritmo de

reconfiguración para conseguir su objetivo: disminuir el tráfico de mensajes que circulan a través de la red y por tanto mejorar las prestaciones del multicomputador. De cara a ver la incidencia de esta técnica en los problemas habituales que se utilizan en cálculo paralelo, en el último punto de este apartado presentaremos los resultados obtenidos para un problema de cálculo matricial.

4.1 Modelo de máquina y presentación de resultados.

Los resultados que vamos a presentar se han obtenido por simulación. Para ello se ha utilizado el entorno FDP [13] que permite simular la conducta de un multicomputador genérico con una topología reconfigurable concreta, por lo cual los resultados obtenidos son fácilmente extrapolables a cualquier otra máquina que permita la reconfiguración de la red de interconexión.

El modelo de máquina que utiliza este entorno (multicomputador genérico) presenta las siguientes características: número de nodos variable a elección del usuario, algoritmo de encaminamiento determinista (e-cube) con una técnica de control de flujo de mensajes del tipo almacenamiento y reenvío, función de asignación (mapeo) de los procesos a los procesadores de forma cíclica y un número de enlaces de comunicación con los otros procesadores para cada procesador limitado a 4, como en el transputer. Esto nos permite la simulación de las topologías más comunes en los multicomputadores como la malla 2-D, el anillo y el hipercubo, este último sólo de grado 4.

Debido a que el mapeo es estático y conocido a través de todo el sistema, el núcleo del *run-time* de un nodo puede determinar si los procesos fuente y destino de un mensaje están en el mismo nodo, o si debe ejecutar el algoritmo de encaminamiento para enviarlo al proceso destino.

La reconfiguración de la red será evaluada por medio de tres parámetros:

1. **Tráfico total de mensajes en la red**. Mide cuantos mensajes cruzan nodos intermedios para ir del nodo origen al nodo destino. Cada vez que un mensaje cruza un nodo intermedio se incrementa el tráfico total de mensajes.

2. **Tráfico máximo en un nodo**. Mide cuantos mensajes han cruzado el nodo más saturado. Este es un parámetro importante ya que un nodo altamente congestionado reduce el rendimiento de todo el sistema.

3. **Número de cambios**. Mide el número de veces que se ha modificado la topología de la red. Debido a que la realización de un cambio conlleva un coste, es importante conocer cuantos cambios se han realizado para obtener un resultado dado. Un excesivo número de cambios podría invalidar los resultados obtenidos en los otros dos parámetros.

Para poder evaluar adecuadamente la mejora obtenida, se compararán los resultados obtenidos con las tres topologías estáticas que simula el FDP (anillo, malla 2-D e hipercubo) junto al resultado obtenido por reconfiguración de la red. Debido a que el algoritmo preserva la topología, en el caso de reconfiguración dinámica hay que elegir una topología base sobre la que trabaje el algoritmo: dicha topología será el hipercubo. Para el caso de ejemplo, se ha tomado un número de nodos igual a 16.

4.2 Caso de ejemplo: dos nodos fuente envían mensajes a un nodo destino.

Para mostrar adecuadamente el funcionamiento de la reconfiguración de la red hemos elegido un sencillo caso en el cual 2 nodos fuente envían mensajes a un nodo destino. En este caso, el nodo 0 y el nodo 2 envían mensajes al nodo 14. Cada nodo envía 100 mensajes al nodo destino. Para la correcta comprensión de esta exposición, es adecuado tener en mente la topología base sobre la que se trabaja: el hipercubo de grado 4, mostrada en la figura 3.

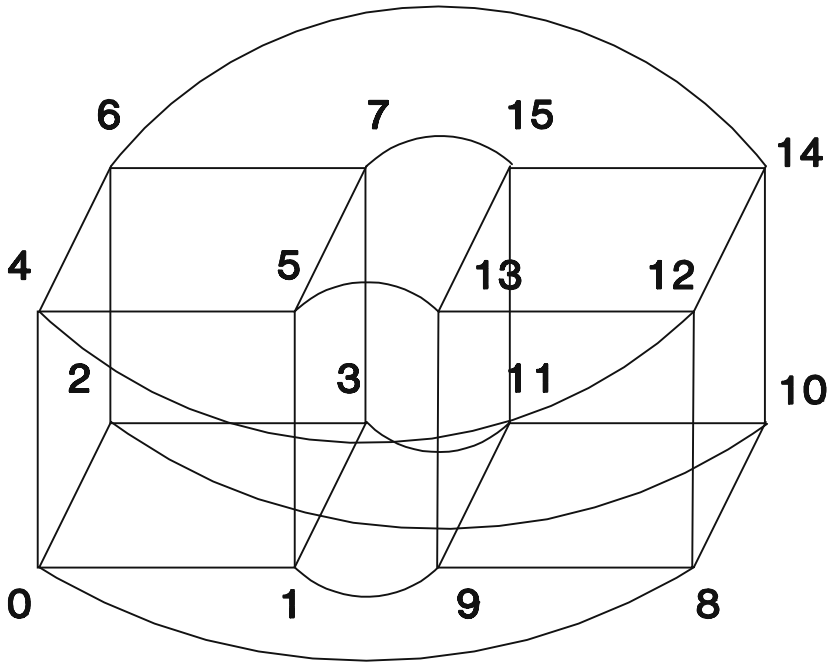

Fig. 3. Hipercubo de grado 4

Nótese la diferencia que hay entre el número que identifica a un nodo y el número que indica la posición que un nodo ocupa en la red de interconexión. Debido a que tenemos 16 nodos (es decir, 16 procesadores físicos), para distinguirlos se numeran de 0 a 15. Inicialmente, al nodo 0 se le asigna la posición 0 de la red de interconexión, al

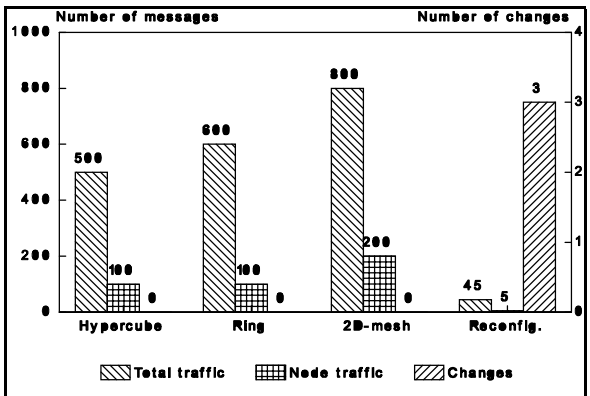

Fig 4. Dos nodos fuente y un destino.

nodo 1 la posición 1, y así, por lo que en la figura 3 los números que aparecen representan tanto las posiciones de la red de interconexión como los nodos físicos que ocupan dichas posiciones. Debido a la posición de los nodos en la red, estos mensajes van a tener que atravesar nodos intermedios provocando un tráfico de mensajes en la red sea cual sea la topología elegida. En el caso de reconfiguración dinámica de la red, este tráfico va a provocar que el algoritmo modifique la topología, acercando los nodos fuente y destino hasta que sean vecinos. En la figura 4 se muestran los resultados obtenidos para las diversas topologías con un comportamiento estático y para el caso de la reconfiguración dinámica. Los valores reflejan los tres parámetros de medida antes citados: el tráfico total en la red, el tráfico máximo en un nodo y el número de cambios realizados en la red. Estos resultados pueden variar ligeramente en función del valor que tomen unos umbrales que posee el algoritmo [14].

Como puede observarse, gracias a la reconfiguración dinámica el comportamiento de la red de interconexión mejora notablemente. El tráfico de mensajes a través de la red prácticamente desaparece (de un valor de 300 -con la mejor topología estática- pasa a un valor de 23), y lo mismo sucede con el tráfico máximo en un nodo singular, todo ello con un bajísimo número de cambios (únicamente 2). En este caso la reconfiguración dinámica de la red permite un aumento de la potencia y del rendimiento del multicomputador, demostrándose lo adecuado de esta técnica para este tipo de máquinas paralelas.

Los cambios realizados en la topología de la red persiguen situar al nodo 14 como vecino de los nodos 0 y 2. Para ello, el algoritmo de reconfiguración que se ejecuta en el nodo 14 provoca un cambio de posición de dicho nodo con

el nodo 6. Posteriormente cambiará la posición con el nodo 2, con lo que habrá conseguido su objetivo: por uno de los enlaces será vecino del nodo 0 (en la posición 0, no se ha movido) y por otro será vecino del nodo 2 (movido a la posición 6), con lo que a partir de ese momento cesará el tráfico de mensajes por los nodos intermedios de la red. En la figura 5 se muestran las posiciones de los diversos nodos en la red de interconexión después de todos estos cambios. En este caso, los números mostrados representan a los nodos físicos. Así, por ejemplo, en dicha figura se puede observar que el nodo 14 ocupa la posición 2 de la red de interconexión.

Una última consideración. Téngase en cuenta que si el número de mensajes enviados fuera mayor (los 100 mensajes considerados es una cifra pequeña), los resultados serían todavía más espectaculares, ya que para una topología estática de la red se obtendría un tráfico mayor, mientras que en el caso de la reconfiguración dinámica el resultado sería idéntico al aquí presentado.

4.3 Resultados para un caso real basado en las rotaciones rápidas de Givens.

Pero estos resultados no se obtienen tan sólo para sencillos casos de ejemplo. A continuación vamos a mostrar los resultados obtenidos para el caso de un típico problema matricial: la triangularización de una matriz dispersa por medio de las rotaciones rápidas de Givens.

Para poder evaluar la reconfiguración dinámica de la red de interconexión se necesita disponer de problemas cuyo patrón de comunicaciones no sea regular y, a ser posible, variable con el tiempo. Para este tipo de problemas no hay ningún modelo de arquitectura de la red que se

adapte adecuadamente, y por tanto se puede explotar la propiedad de la reconfiguración dinámica de la red.

Junto a la importancia científica del algoritmo escogido [11], el factor decisivo que nos ha movido a elegir este ejemplo es que no se puede conocer a priori el modelo de comunicación que se va a dar entre los diferentes nodos, debido a que dependerá de la estructura de la matriz vacía que se va a triangularizar. Por tanto en este algoritmo es difícil realizar un adecuada distribución (mapeo) de los procesos en los procesadores de cara a minimizar el coste de las comunicaciones entre nodos. Más aún, el modelo de comunicación variará a lo largo del tiempo conforme se vayan procesando las filas. Por ello, va a ser un algoritmo adecuado para mostrar las ventajas que presenta la reconfiguración dinámica de la red.

En el caso de matrices vacías es difícil analizar teóricamente la complejidad de las comunicaciones, ya que la estructura de la matriz cambia durante el procesamiento. Así pues, el estudio de prestaciones que se va a mostrar en este capítulo ha sido experimental y realizado con el FDP.

Las matrices de prueba para el análisis del algoritmo de Givens se han generado de forma aleatoria realizando una distribución homogénea de los elementos no nulos en la matriz.

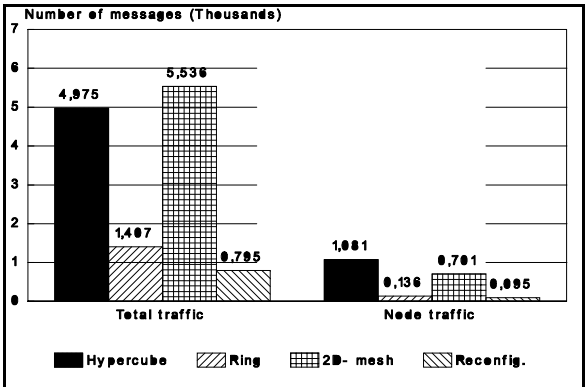

Fig. 6. Resultados para una matriz de 600x300.

Una explicación más detallada del algoritmo de Givens así como de las principales características de las matrices se puede encontrar en [11]. A modo de ejemplo, se muestra en la figura 6 los

resultados obtenidos para una matriz vacia de tamaño 600x300. En dicha gráfica, se muestra el tráfico de mensajes por la red para las tres topologías antes citadas, así como el tráfico de mensajes obtenido por medio de la reconfiguración dinámica de la red. Como los resultados de la reconfiguración dependen de unos umbrales [14], se muestra el mejor valor obtenido así como el valor medio. Dichos resultados confirman lo adecuado que resulta la reconfiguración dinámica de la red de interconexión para la reducción del tráfico de mensajes por la red, y por lo tanto, para la mejora en el rendimiento de los multicomputadores. La reducción del tráfico total en la red de interconexión es drástica, obteniéndose para la mejor pareja de umbrales unos valores del 15-20% del tráfico que se obtiene en el caso de una topología estática de hipercubo, y unos valores de un 60-65% con respecto a una topología estática de anillo. Conforme el tamaño de la matriz aumenta los resultados son mejores.

5. TRABAJO RELACIONADO.

Por último, y antes de finalizar este artículo, nos parece interesante comentar las características de los trabajos que se han realizado en este campo. Son los siguientes:

a) **El proyecto europeo Esprit P1085**. Dicho proyecto comenzó en diciembre de 1985 con el objetivo de *desarrollar una máquina multiprocesador con un alto rendimiento y un bajo coste*. Dicho proyecto ha sido principalmente desarrollado entre diversas universidades y centros de investigación de Francia e Inglaterra. La característica clave del proyecto P1085 era la idea de que un computador numérico de propósito general podía ser construido a base de una red reconfigurable de transputers, pudiendo los programadores

usar esta máquina de forma efectiva para sus aplicaciones habituales, utilizando el lenguaje Occam. El éxito de este proyecto puede medirse por la explotación comercial de dicho diseño: han habido dos casas comerciales -Telmat y Parsys- que han sacado al mercado máquinas basadas en este diseño. En [16] se puede encontrar una explicación bastante detallada de los objetivos de dicho proyecto y de los logros alcanzados, así como de la arquitectura que presenta dicha máquina.

Con respecto a la reconfiguración dinámica de la red, se han podido encontrar pocas referencias. Quizás la más interesante sea la encontrada en [1]. En ese artículo se describe un interfaz de alto nivel el cual sirve para desarrollar aplicaciones con topologías variables en un Supernodo. Este trabajo es adecuado para problemas cuyo patrón de comunicaciones varía con el tiempo pero *a priori* ya se sabe cual es la topología más adecuada en cada una de las fases. Entonces entre cada fase se insertan puntos de reconfiguración cuyo objetivo es modificar la topología para adecuarla a los requerimientos de esa fase.

b) **El sistema DAMP**. En la Universidad de Paderborn se desarrolló un sistema multicomputador con una red de interconexión reconfigurable dinámicamente. Su arquitectura se basa en un bloque básico (el módulo DAMP) que consiste en un transputer con su memoria local y un interfaz para la conexión con otros módulos básicos. Estos módulos básicos se conectan de acuerdo a una topología fija con un número limitado de vecinos (lo máximo es un toroide octogonal, es decir, ocho vecinos). El interfaz de conexión que tiene cada bloque básico permite modificar la topología de interconexión de forma dinámica durante la ejecución del programa. Actualmente hay desarrollado un prototipo con ocho procesadores y se está trabajando en uno con 64

procesadores. En [3] se describe la arquitectura básica de este sistema así como las propiedades de la reconfiguración de la red (especialmente los problemas de bloqueo). Asimismo, se trata acerca del tipo de control adecuado para dicha reconfiguración (centralizado o descentralizado) y se dan las primeras medidas obtenidas acerca de los tiempos de reconfiguración con dicho sistema.

c) **Otros trabajos**. En California hay un grupo que está trabajando con redes de transputers con reconfiguración dinámica. En [15] se proponen y estudian las tres formas para realizar la reconfiguración: con topología fija, con múltiples topologías y con topología aleatoria. Se analizan dichas tres formas y se dan unos primeros resultados obtenidos.

Otro trabajo del que tenemos noticia es el que se está desarrollando en el Laboratorio de Informática del Paralelismo en la Universidad de Lyon [7]. Usando una máquina Tnode (desarrollada por Telmat) están estudiando el problema de la reconfiguración dinámica de la red aplicado a mejorar el rendimiento de dicho multicomputador en problemas de álgebra lineal.

6. CONCLUSIONES.

En este artículo se ha presentado el estudio de una característica novedosa de los multicomputadores: **la reconfiguración dinámica de la red de interconexión**. Nuestro interés surgió al comprobar como el principal problema que presentan los multicomputadores es debido al retraso introducido en las comunicaciones entre los diversos procesos. Una solución alternativa a las que tradicionalmente se vienen utilizando para aliviar este problema es explotar una nueva característica arquitectónica de estas máquinas, que es la posibilidad de reconfigurar dinámicamente la topología de

la red de interconexión entre los diversos procesadores.

Para cubrir este objetivo se ha estudiado la red de interconexión desarrollándose diversos conceptos importantes relativos a la reconfiguración dinámica. Junto a esto, y teniendo en mente una máquina real basada en transputers en la que implementar este trabajo -el Parsys SN 1000-, se ha explicado el protocolo que hemos desarrollado para llevar a cabo la reconfiguración dinámica de la red. Dicho protocolo se ha desglosado en una parte para el *system controller* y otra parte para el resto de nodos del sistema. Para realizar este protocolo es necesario que en cada nodo haya un algoritmo que se encarge de la reconfiguración de la red. Por ello se ha desarrollado un algoritmo que permita *decidir* cuando es oportuno llevar a cabo una reconfiguración, y realice los pasos necesarios para cambiar la topología.

Dicho protocolo se ha explicado por medio de un ejemplo sencillo, donde se ha ilustrado el modo de funcionamiento del algoritmo de reconfiguración así como los buenos resultados obtenidos. Asimismo, se presentan resultados obtenidos a partir de un algoritmo matricial, la triangularización de una matriz vacia por medio de las rotaciones rápidas de Givens. Todos estos resultados demuestran que la reconfiguración dinámica de la red es una característica muy adecuada para aumentar el rendimiento y la potencia de los multicomputadores.

Agradecimientos

Este trabajo ha sido parcialmente financiado por la Comisión Interministerial de Ciencia y Tecnología (CICYT), Proyecto No. TIC91-1157-C03-03.

REFERENCIAS

[1] Adamo, J. and Bonello, C. "Tenor++: A dynamic configurer for Supernode machines". Lecture Notes in Computer Science No. 457, pp. 640-651, Springer-Verlag, 1990.

[2] Athas, W.C. and Seitz, C.L. "Multicomputers: Message-passing concurrent computers". IEEE Computer, Vol. 21, No. 8, pp. 9-24, August 1988.

[3] Bauch, A.; Braam, R. and Maehle, E. "DAMP: A dynamic reconfigurate multiprocessor system with a distributed switching network". 2nd European Distributed Memory Computing Conference, Munich, April 1991.

[4] Bokhari, S.H. "On the mapping problem". IEEE Trans. on Computers, Vol. C-30, No. 3, pp. 207-214, March 1981.

[5] Dally, W.J. and Seitz, C.L. "The torus routing chip". Distributed Computing, Vol. 1, No. 3, pp. 187-196, October 1986.

[6] Dally, W.J. and Seitz, C.L. "Deadlock-free message routing in multiprocessor interconnection networks". IEEE Trans. Computers, Vol. C-36, No. 5, pp. 547-553, May 1987.

[7] Desprez, F. et Tourancheau, B. "Evaluation des performances de la machine Tnode". La lettre du Transputer, LIB Besacon. France. No. 7, 1990.

[8] Duato, J. "On the design of deadlock-free adaptive routing algorithms for multicomputers: design methodologies". Parallel Architectures and Languages Europe 91, Eindhoven, June 1991.

[9] Duato, J. "A new theory of deadlock free adaptive routing in wormhole networks". IEEE Trans. on Parallel and Distributed Systems. Vol. 4, No. 12, pp. 1320-1331, December 1993.

[10] Feng, Tse-yun. "A Survey of interconnection networks". IEEE Computer 14, pp. 12-27, December 1981.

[11] García, J.M. and Duato, J. "An algorithm for dynamic reconfiguration of a multicomputer network". Third IEEE Symposium on Parallel and Distributed Processing, Dallas (Texas), December 1-5, 1991.

[12] García, J.M. and Duato, J. "Evaluating the cost of the dynamic reconfiguration of a multicomputer network", Actas de la XVIII Conferencia Latinoamericana de Informática PANEL'92, pp. 523-530, Las Palmas de Gran Canaria, Septiembre, 1992.

[13] García, J.M. and Duato, J. "An advanced environment for programming transputer networks with dynamic reconfiguration", in M. Valero, E. Oñate, M. Jane, J.L. Larriba and B. Suárez (Eds.) Parallel Computing and Transputers Applications, IOS Press/CIMNE, pp. 601-610, 1992.

[14] García, J.M. and Duato, J. "Dynamic reconfiguration of multicomputer network: Limitations and Tradeoffs", in P. Milligan and A. Nuñez (Eds.), Euromicro Workshop on Parallel and Distributed Processing, IEEE Computer Society Press, pp. 317-323, 1993.

[15] Jin, L. et al. "Dynamically reconfigurable architecture of a transputer-based multicomputers system". 20 Int. Conf. on Parallel Processing, August, 1991.

[16] Nicole, D.A. "Reconfigurable Transputer Processor Architectures", in Multi-processor Computer Architectures, T.J. Fountain and M.J. Shute (Editors), North-Holland, 1990.

[17] Nicol, D.M. and Reynolds, P.F.Jr. "Optimal dynamic remapping of data paralell computations". IEEE Trans. on Computers, Vol C-39, No. 2, pp. 206-219, February 1990.

[18] Rattner, J. "The new age of supercomputing" Proc. 2nd European Distributed Memory Computing Conference, Munich, FRG, April 1991.

[19] Seitz, C.L. "The Cosmic Cube". Comm. of the ACM 28, No. 1, pp. 22-33, January 1985.

[20] Tudruj, M. and Thor, M. "The architecture of a multilayer dynamically reconfigurable transputer system". 20 Int. Conf. on Parallel Processing, August, 1991.

[21] Ward, J.S.; Roberts, J.B.G. and Simpson, P. "Optimizing a reconfigurable transputer array for line-of-sight communications". In Aspects of Computation on Asynchronous Processors, M. Wright (Ed.), 223-224, North-Holland 1989.

[22] Whitby-Strevens, C. "The transputer". 12th Int. Symp. on Computer Architecture, Boston, June 1985.