Evaluación de entornos de tráfico real de una arquitectura de conmutador con soporte para calidad de servicio y eliminación del HOL blocking *

J. Villar, P.J. García, F.J. Alfaro
J.L. Sanchez, F.J. Quiles
U. de Castilla–La Mancha
{juanan,pgarcia,falfaro,jsanchez,paco}@dsi.uclm.es

A. Martínez
Intel Barcelona Research Center
AlejandroX.Martinez@intel.com

J. Flich, J. Duato
U. Politécnica de Valencia
{jflich,jduato}@disca.upv.es

Resumen

La red de interconexión es un elemento esencial de cualquier sistema de computación paralela, desde supercomputadores hasta servidores de Internet. En la actualidad, una de las principales preocupaciones de los diseñadores de redes de interconexión de altas prestaciones es aumentar el rendimiento de la red, empleando la mínima cantidad de recursos. En este sentido, en este artículo se describe y evalúa una arquitectura de conmutador que utiliza los mismos recursos para proporcionar dos aspectos clave para el rendimiento de la red: garantías de QoS y control de congestión. En la evaluación se compara esta arquitectura con otras utilizadas habitualmente en el mismo contexto. Los resultados en los que se basa la evaluación se han obtenido mediante simulación, empleando como carga de tráfico la captura de tráfico real producido por el benchmark LINPACK. A partir de los resultados, se puede concluir que en escenarios de tráfico real, nuestra propuesta es tan efectiva como las anteriores, mientras que requiere una cantidad menor de recursos, lo que la hace económicamente más viable.

*Este trabajo ha sido cofinanciado por el M.E.C. y M.I.C.I.N.N. del gobierno de España, y por fondos FEDER de la Comisión Europea, con las subvenciones "Consolider Ingenio–2010 CSD2006-00046" y "TIN2009–14475–C04", respectivamente; por la Consejería de Educación y Ciencia de Junta de Comunidades de Castilla–La Mancha con los proyectos "PCC08–0078–9856", "POII10–0289–3724" y con una Beca Predoctoral de Investigación "07/096".

1. Introducción

Las redes de interconexión se han convertido en un elemento clave en muchos tipos de sistemas de computación paralela, puesto que deben proporcionar la latencia y productividad demandadas por las actuales aplicaciones paralelas. Debido a la proliferación de estos sistemas, el interés de los investigadores por los mismos ha aumentado, siempre con la intención de mejorar el rendimiento de la red. Además, el coste de los componentes de red, su consumo, la disipación de calor, etcétera, se han convertido en factores que determinan la viabilidad económica de los diseños. De esta forma, determinadas propuestas que resultaban válidas en el pasado para mejorar las prestaciones de la red presentan ahora graves problemas arquitectónicos de escalabilidad, haciendo imposible su empleo en sistemas más modernos.

Por ejemplo, en el pasado se evitaba que la red llegara a situaciones de congestión sobredimensionando su tamaño, lo que garantizaba que la red no sufriría la degradación de prestaciones propia de estas situaciones. Esta degradación es debida al bloqueo de cabeza de línea (*Head–Of–Line*, HOL blocking), que sucede cuando un paquete en cabeza de una cola FIFO se bloquea debido a la congestión e impide que avancen otros paquetes almacenados en la misma cola, incluso si éstos solicitan recursos disponibles. El HOL blocking puede degradar drásticamente las prestaciones de la red, puesto que puede llevar a que los flujos de

paquetes que no contribuyen a la congestión avancen al mismo ritmo que los flujos congestionados.

Obviamente, sobredimensionar la red es actualmente inviable, por lo que se han propuesto técnicas más elaboradas específicamente orientadas a solucionar los problemas relacionados con la congestión (ver sección 2). En general, dichas técnicas mejoran las prestaciones de la red por cuanto impiden la degradación del rendimiento en situaciones de congestión. Muchas de ellas se basan en separar los flujos de paquetes en colas distintas evitando así el HOL blocking o reduciéndolo.

Otra forma de mejorar el rendimiento de la red, desde el punto de vista de las aplicaciones, es usar técnicas que ofrecen garantías de calidad de servicio (*Quality of Service*, QoS). En general, si la tecnología de red puede segregar el tráfico en clases, la QoS consiste en garantizar un mínimo rendimiento a cada clase de tráfico, con respecto al resto de clases de tráfico. Por ejemplo, se puede garantizar un porcentaje del ancho de banda a una clase, y se debe asegurar que la clase recibe dicho ancho de banda incluso si en la red se está inyectando más tráfico del admitido.

La solución habitual para garantizar QoS es utilizar un canal virtual (CV) [5] para el tráfico de cada clase. El control de flujo se gestiona a nivel de CV, esto es, hay un contador individual de créditos para cada CV. De este modo, se logran dos objetivos: se elimina el HOL blocking entre paquetes de diferentes CVs, y los paquetes de una clase de tráfico no pueden ocupar todo el espacio disponible del buffer. En consecuencia, el rendimiento de cada clase de tráfico sólo depende de los algoritmos de planificación implementados en los conmutadores y de la cantidad de tráfico inyectado en el red.

Como puede comprobarse, el planteamiento de las técnicas de control de congestión basadas en eliminar el HOL blocking y el planteamiento de las técnicas de QoS basadas en CVs es similar, al emplearse en ambos casos colas distintas para separar distintos flujos. Por tanto, no es extraño plantear una arquitectura de conmutador que emplee los mismos recursos para proporcionar ambos aspectos. En este sentido, en este trabajo se presentan los resultados de la evaluación en entornos de tráfico real de una arquitectura eficiente de conmutador con soporte de calidad de servicio y control de congestión mediante la eliminación del HOL blocking. Esta arquitectura ha sido presentada y evaluada mediante simulación con tráfico sintético en [14]. En esta ocasión la eficiencia y escalabilidad de esta arquitectura se han evaluado mediante la utilización del tráfico que se ha obtenido mediante la captura del tráfico de red producido por el benchmark LINPACK en un sistema real.

El resto del artículo se estructura del siguiente modo: en las secciones 2 y 3 se repasa la problemática del control de congestión en las redes de interconexión y algunas de las propuestas existentes para garantizar calidad de servicio, respectivamente. La sección 4 se centra en explicar brevemente el funcionamiento de nuestra propuesta. En la sección 5 se presentan los resultados de la evaluación de la técnica mediante las trazas utilizadas. Finalmente, la sección 6 presenta las conclusiones.

2. El problema de la congestión en redes de interconexión

El fenómeno de la congestión en redes de interconexión es un problema ampliamente conocido, y existen numerosas propuestas para solucionarlo. Las estrategias más simples son el sobredimensionamiento de la red y el descarte de paquetes en situaciones de congestión. Sin embargo, ninguna de ellas se adecúa a las características de las redes de interconexión modernas debido al alto coste y consumo de potencia de los actuales componentes de red, y a la filosofía de transmisión "sin pérdidas" (es decir, sin descarte de paquetes) propia de estas redes.

Tradicionalmente, la forma más eficaz de minimizar la congestión de red ha sido la utilización de colas virtuales (*Virtual Output Queues*, VOQ) [4]. Según esta propuesta en cada puerto del conmutador existen tantas colas como destinos en la red, y un paquete se almacena en la cola asignada al destino de di-

cho paquete. De este sencillo modo, se logra eliminar el HOL blocking entre paquetes con destinos diferentes. No obstante, esta técnica no es escalable con el tamaño de la red, pues la cantidad de colas en cada puerto del conmutador puede llegar a ser muy alta, por lo que el área de silicio necesaria para albergar los buffers que implementan las colas dispara el coste final.

Una alternativa más viable consiste en seguir utilizando VOQ, pero empleando en cada puerto tantas colas como puertos tiene el conmutador [2]. Esto reduce considerablemente los requisitos de silicio. Sin embargo, esta estrategia sólo elimina el HOL blocking a nivel de conmutador, pero no es suficiente para eliminar el HOL blocking a nivel de red.

A diferencia de las estrategias anteriores, RECN (*Regional Explicit Congestion Notification*) [7, 9] elimina por completo el HOL blocking utilizando una cantidad pequeña de recursos, independiente del tamaño de la red, lo que hace a RECN una técnica realmente eficiente y escalable. Para conseguirlo, RECN detecta la existencia de congestión y asigna dinámicamente colas separadas para contener los paquetes pertenecientes a cada flujo congestionado. Respecto a los paquetes de los flujos que no generan congestión, RECN asume que éstos pueden mezclarse en la misma cola sin que se produzca HOL blocking de forma significativa. De esta manera, RECN consigue mantener las prestaciones de la red incluso en presencia de congestión.

En cuanto a sus requisitos, RECN precisa el uso de encaminamiento fuente determinista de cara a identificar un punto concreto de la red desde cualquier otro punto. De hecho, RECN ha sido diseñado para PCI Express Advanced Switching (AS) [1], una tecnología que emplea encaminamiento fuente[1]. La cabecera de los paquetes AS contiene un campo (*turnpool*) compuesto por 31 bits que codifican la secuencia de desplazamientos que un paquete debe realizar entre la entrada y la salida de cada conmutador de su ruta. Al usarse este sistema, un conmutador puede averiguar por anticipado si un paquete entrante pasará por un determinado punto de la red.

De cara a separar los flujos congestionados de los no congestionados, RECN añade a la cola común de cada puerto un conjunto de colas adicionales (*Set Aside Queues*, SAQs). La cola estándar se emplea para almacenar paquetes no congestionados, mientras que las SAQs se asignan dinámicamente para almacenar paquetes que pasarán por puntos de congestión. Las SAQs de cada puerto se controlan mediante una memoria CAM (*Content Addressable Memory*). Cada uno de los registros en los que se divide la CAM contiene la información necesaria para identificar un punto de congestión y para gestionar la SAQ asignada a dicho punto.

RECN también detecta la desaparición de congestión en cualquier punto, de manera que las SAQs asignadas a dicho punto pueden liberarse y asignarse posteriormente a nuevos puntos de congestión. Esto permite a RECN eliminar el HOL blocking usando un número muy reducido de SAQs.

3. Soporte de QoS en redes de interconexión

En algunas tecnologías de interconexión, como InfiniBand o PCI Advanced Switching, la estrategia para proporcionar garantías de QoS consiste en proporcionar canales virtuales separados a cada clase de tráfico (*service level*, SL). Esto aumenta la complejidad del conmutador y área de silicio, por lo que muy pocas implementaciones proveen todos los CVs propuestos por las especificaciones. InfiniBand [11] soporta hasta 16 CVs y la tecnología PCI Advanced Switching define hasta 20 CVs.

A fin de poder aliviar estos problemas se han propuesto algunas técnicas para reducir la cantidad de CVs y seguir proporcionando garantías de QoS. Por ejemplo, [12] propone una técnica para proporcionar QoS con sólo dos CVs. Esta técnica es más simple y general que las anteriores. La idea es planificar el tráfico en las interfaces de red de modo que los conmutadores asumen que el orden de recep-

[1] Sin embargo, conviene dejar claro que RECN podría aplicarse a cualquier otra tecnología siempre que ésta utilice encaminamiento fuente determinista.

ción de los paquetes significa implícitamente la prioridad y ellos simplemente se esforzarán en mantenerla.

Por otra parte, ha habido propuestas que intentan ofrecer QoS con tan sólo dos CVs. Por ejemplo, el Avici TSR [4] es capaz de segregar el tráfico en dos categorías, una preferente y otra normal. Lamentablemente, está limitada a esta clasificación de tráfico y no puede diferenciar más categorías. La arquitectura propuesta en [3] asigna múltiples clases de tráfico a dos colas. Sin embargo, esta propuesta no escala y sólo funciona en un conmutador de una sola etapa basado en un único crossbar. Dicho crossbar utiliza buffers en los puntos de cruce que se dividen en dos CVs.

En [13] la técnica anterior se combina con RECN. Las estructuras de las colas de RECN se duplican para dos CVs, a fin de obtener un conmutador capaz de proporcionar al mismo tiempo garantías de QoS y gestión de congestión.

En [14] se proporciona una versión que supera las prestaciones del mecanismo básico de RECN de modo que proporciona QoS sin la utilización de CVs adicionales. Esta arquitectura toma en consideración el paralelismo existente entre las tareas propias de RECN y el uso de CVs empleados para proporcionar QoS.

4. Arquitectura del conmutador

Nuestra propuesta consiste en una arquitectura de conmutador capaz de soportar QoS y control de congestión, con recursos reducidos. La organización de un conmutador consiste en una combinación de buffers en los puertos de entrada y de salida. Todos ellos deben implementarse en un único chip, para conseguir las bajas latencias de cruce demandadas por las aplicaciones paralelas.

La organización de un puerto de entrada puede verse en la figura 1. Este es el esquema estándar para un puerto en RECN, de modo que habrá tantas colas de detección (*Detection Queues*) como puertos tiene el conmutador. Las colas de detección almacenan los paquetes de los flujos no congestionados. También existen colas SAQs que contienen los paquetes de flujos congestionados. Por otra parte, en la figura 2 puede verse la organización de la CAM. La CAM es necesaria en los puertos de entrada, y de salida, para poder gestionar las SAQs. La estructura de nuestra CAM es similar a la de RECN, salvo que nuestra propuesta añade un campo adicional por cada fila. El nuevo campo indica el SL al que se ha asignado la SAQ.

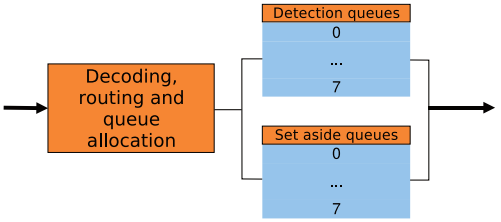

Figura 1: Organización del puerto de entrada.

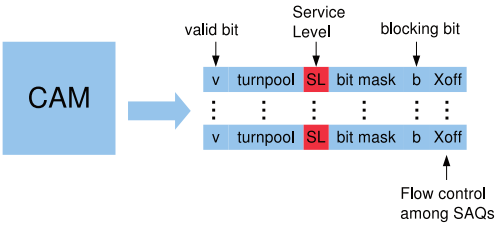

Figura 2: Organización de la CAM.

La figura 3 muestra la organización del puerto de salida. En este caso las colas de detección no son necesarias por lo que únicamente se utiliza una cola estándar para almacenar los paquetes no congestionados. En base al esquema de RECN, aquí también se utilizan colas SAQs. Adicionalmente, nuestra propuesta implementa contadores de ancho de banda en el puerto de salida, uno por cada SL. Estos contadores deben calcular dinámicamente la diferencia entre el ancho de banda reservado y consumido. Los contadores se utilizan por dos razones: se utilizan en el planificador (*Deficit Round Robin*, DRR) del conmutador, y por el control de congestión.

El planificador DRR consiste en dar prioridad a los paquetes que están consumiendo

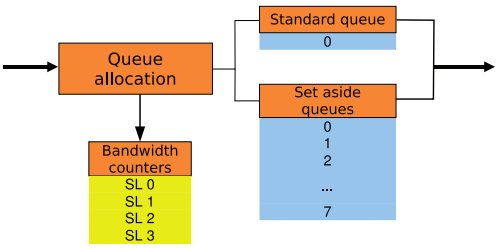

Figura 3: Organización del puerto de salida.

menos ancho de banda del que se les reservó en principio. Los paquetes de flujos congestionados son penalizados por el planificador DRR. RECN detecta la congestión midiendo el número de paquetes almacenados en las colas de los puertos. Nuestra propuesta detecta la congestión igual que RECN, y además, utiliza los contadores de ancho de banda. En general, cuando nuestra propuesta detecta que la ocupación en las colas y los contadores de ancho de banda han superado unos umbrales preestablecidos, entonces los paquetes congestionados se mueven a una cola SAQ, de modo que se consigue aislar los flujos congestionados de los no congestionados. Una vez que la congestión desaparece, los flujos dejarán libre la SAQ que les fue asignada, y vuelven a sus respectivas colas estándares.

5. Evaluación de prestaciones

A continuación, se presenta una nueva evaluación de rendimiento de la arquitectura propuesta en [14], y que ahora se extiende con la utilización de tráfico real.

5.1. Arquitectura simulada

El tráfico de red se distribuye entre cuatro SLs con prioridad decreciente, tal que el SL 0 tiene la mayor prioridad y el SL 7 la menor. No obstante, a cada SL se le garantiza un 25 % del ancho de banda de cada enlace.

La topología de red utilizada para las pruebas es una red indirecta multietapa bidireccional de diferentes tamaños: 8, 16, 64 y 256 puertos. Para la conexión entre las etapas de la red se ha utilizado la permutación *butterfly* [8]. Se ha escogido una multietapa porque es una topología habitual para clusters de PCs y otras redes de interconexión. Sin embargo, nuestra propuesta es válida para cualquier otra topología, incluyendo redes directas.

Se han considerado cuatro arquitecturas de conmutador:

VOQsw + 1CV. Es el clásico VOQ a nivel de conmutador y un único CV que acoge el tráfico de los cuatro SLs.

VOQsw + 4CVs. Similar al anterior, pero con 4 CVs. El tráfico de cada SL se asigna a un CV distinto.

VOQnet + QoS. Es la técnica de VOQ a nivel de red con un CV por cada SL. Es la arquitectura ideal, pero no es implementable porque utiliza demasiados recursos. Se utiliza habitualmente para hacer comparaciones.

RECN + QoS. Es la arquitectura que integra control de congestión y garantías de QoS que se desea evaluar.

Para todas las arquitecturas, los conmutadores tienen 8 puertos. El tamaño máximo de paquete es de 4 Kbytes y el ancho de banda de los enlaces es 10 Gb/s. Otra suposición es el uso de encaminamiento fuente.

5.2. Modelo de tráfico

Para todas las pruebas, se han utilizado trazas reales obtenidas en un cluster de 32 nodos y 8 *cores* por nodo. Cada nodo dispone de 32 Gbytes de RAM. La topología de la red del cluster es un *fat–tree*. La topología real del cluster donde se ha capturado la traza no es relevante a fin de realizar esta evaluación puesto que el tiempo guardado en la traza no se toma en consideración, sino que sólo se utiliza la relación causal existente entre el envío y la recepción de los mensajes MPI guardados en la traza.

Para la obtención de las trazas se ha modificado la librería MPICH siguiendo las instrucciones de [16]. En concreto, se ha modificado el módulo MPE (*Multi Processing Environment*)

de MPICH para que la traza registre todas las comunicaciones, incluso las que implementan las operaciones colectivas y que por defecto no quedan registradas [17].

La traza contiene los mensajes capturados por la librería mientras se ejecuta el LINPACK [6, 15]. El LINPACK es un benchmark que resuelve un sistema de ecuaciones lineales. Se utiliza para formar la lista de los 500 supercomputadores más potentes del mundo [18]. El rendimiento alcanzado por el benchmark depende de varios parámetros. El benchmark se ha configurado para que ocupe 2GBytes de RAM por proceso MPI. El resto de parámetros se han ajustado para que la traza no ocupe mucho espacio en disco y al mismo tiempo la cantidad de mensajes sea significativa.

El tráfico LINPACK se inyecta por el SL 0 y el resto de SLs contienen tráfico de ruido. El tráfico de ruido se genera a partir de la traza del sistema de E/S Cello de Hewlett–Packard [10]. Cada SL (1, 2 y 3) transporta una copia de la traza Cello de forma indefinida. A fin de que el tráfico Cello se distribuya por todos los nodos, el origen y destino leídos de la traza Cello se barajan entre el número de nodos de la red. Además, debido a la antigüedad de las trazas se ha aplicado un factor de compresión a la traza Cello para que produzca congestión en la red: 0.75, 1.5, 6 y 24 en función del tamaño de red.

Se ha asumido que el tiempo de procesamiento de paquetes en los nodos de la red es cero, a fin de congestionar más la red y obtener una estimación de rendimiento más restrictiva, puesto que en un entorno realista los nodos consumirán tiempo en computar los paquetes, y la utilización de la red será menor. Sin embargo, la relación causal entre los mensajes se mantiene, pues un nodo sólo enviará mensajes si antes ha recibido todos los mensajes dependientes desde el resto de nodos.

Para la evaluación de las arquitecturas, sólo se ha medido el tiempo de ejecución. Concretamente, para cada arquitectura se ha medido el número total de ciclos de red transcurridos, desde que se envía el primer paquete, hasta que se recibe el último paquete de la traza LINPACK.

En la figura 4 se muestran los resultados de las simulaciones, que para cada tamaño de red muestran el tiempo de ejecución normalizado respecto al caso de referencia (VOQsw+1CV). Las arquitecturas VOQsw+4CV y VOQnet+QoS alcanzan un rendimiento bastante pobre respecto a VOQsw+1CV en redes pequeñas, como es de esperar. Cuando el tamaño de la red alcanza los 256 nodos, VOQsw+1CV sufre una caída del 75 % en el rendimiento respecto al resto de arquitecturas. En todas las redes, el rendimiento de RECN+QoS es excelente, y simplemente utiliza un CV para procesar el tráfico de todas las clases de tráfico.

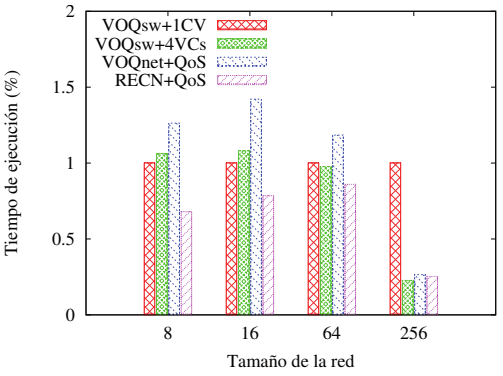

Figura 4: Tiempo normalizado para cada tamaño de red.

También es interesante estudiar cómo escala nuestra propuesta. En la figura 5 puede comprobarse cómo nuestra técnica escala con el tamaño de red. Todos los valores, de todos los tamaños de red se muestran normalizados con respecto al caso VOQsw+1CV de la red de 8 nodos. Obviamente, se observa que el tiempo de ejecución aumenta con el tamaño de red, pero mientras VOQsw+1CV aumenta exponencialmente en la red de 256, las técnicas restantes, incluida nuestra propuesta se mantienen en el mismo orden de magnitud que la red de tamaño 64.

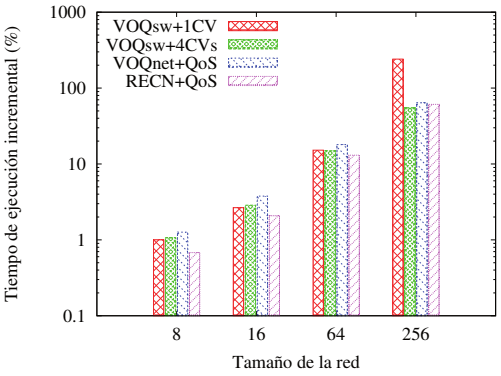

Figura 5: Tiempo normalizado incremental respecto VOQsw+1CV en la red de tamaño 8.

6. Conclusiones

Las redes de interconexión actualmente están demandando garantías de calidad de servicio y control de congestión para alcanzar el rendimiento de red requerido por las aplicaciones paralelas. En este artículo, se ha reevaluado una propuesta de arquitectura de conmutador que es capaz de garantizar calidad de servicio y proporcionar control de congestión, de un modo eficiente puesto que utiliza una cantidad mínima de recursos, es más, utiliza menos recursos que las técnicas de VOQ tradicionales, ya que no utiliza canales virtuales. La arquitectura fue evaluada en trabajos previos con tráfico sintético, y en este trabajo se aporta una nueva evaluación con tráfico real de la ejecución del benchmark LINPACK en un cluster.

Los resultados han mostrado que la arquitectura sigue proporcionando calidad de servicio al tráfico del benchmark LINPACK, eliminando la congestión creada por el tráfico de ruido que existe en la red.

También se ha visto que nuestra arquitectura escala adecuadamente con el tamaño de red. Mientras que VOQ a nivel de switch ha aumentado en un orden de magnitud el tiempo de ejecución, nuestra técnica se ha mantenido en un tiempo similar al que tiene en otras redes de menor tamaño, pero utilizando una cantidad mínima de recursos.

Referencias

[1] Advanced Switching Interconnect Special Interest Group. *Advanced Switching Core Architecture Specification. Revision 1.1*, March 2005.

[2] T. Anderson, S. Owicki, J. Saxe, and C. Thacker. High-speed Switch Scheduling for Local-area Networks. *ACM Transactions on Computer Systems*, 11(4):319–352, November 1993.

[3] N. Chrysos and M. Katevenis. Multiple Priorities in a Two-lane Buffered Crossbar. In *Proceedings of the IEEE GLOBECOM 2004 Conference*, November 2004. CR-ROM paper ID "GE15-3".

[4] W. Dally, P. Carvey, and L. Dennison. Architecture of the Avici Terabit Switch/Router. In *Proceedings of the 6th Symposium on Hot Interconnects*, pages 41–50, 1998.

[5] W.J. Dally. Virtual-channel Flow Control. *IEEE Trans. Parallel Distrib. Syst.*, 3(2):194–205, 1992.

[6] J. Dongarra. Performance of Various Computers using Standard Linear Equations Software. Technical Report Computer Science Technical Report Number CS-89-85, University of Tennessee, Knoxville TN, 37996. `www.netlib.org/benchmark/performance.ps`.

[7] J. Duato, I. Johnson, J. Flich, F. Naven, P.J. García, and T. Nachiondo. A New Scalable and Cost-effective Congestion Management Strategy for Lossless Multistage Interconnection Networks. In *Proceedings of the 11th Symposium on High Performance Computer Architecture (HPCA)*, 2005.

[8] J. Duato, S. Yalamanchili, and N. Lionel. *Interconnection networks. An Engineering Approach.* Morgan Kaufmann Publishers Inc., 2002.

[9] P.J. García, J. Flich, J. Duato, I. Johnson, F.J. Quiles, and F. Naven. Dynamic Evolution of Congestion Trees: Analysis and Impact on Switch Architecture. *Lecture Notes in Computer Science (HiPEAC 2005)*, 3793:266–285, November 2005.

[10] Hewlett-Packard Corporation. SSP Homepage. `http://ginger.hpl.hp.com/research/itc/cls/ssp`.

[11] InfiniBand Trade Association. *InfiniBand Architecture Specification Volume 1. Release 1.0*, October 2000.

[12] A. Martínez, F.J. Alfaro, J.L. Sánchez, and J. Duato. Providing Full QoS Support in Clusters using only Two VCs at the Switches. In *Proceedings of the 12th International Conference on High Performance Computing (HiPC)*, pages 158–169, December 2005.

[13] A. Martínez, P.J. García, F.J. Alfaro, J. Flich, J.L. Sánchez, F.J. Quiles, and J. Duato. A Cost-effective Interconnection Architecture with QoS and Congestion Management Support. August 2006.

[14] A. Martínez, P.J. García, F.J. Alfaro, J.L. Sánchez, J. Flich, F.J. Quiles, and J. Duato. A Switch Architecture Guaranteeing QoS Provision and HOL Blocking Elimination. *IEEE Trans. Parallel Distrib. Syst.*, 20(1):13–24, 2009.

[15] Antoine Petitet, Clint Whaley, Jack Dongarra, and Andy Cleary. LINPACK User's Guide `www.netlib.org/benchmark/hpl`

[16] F.J. Ridruejo, J.Navaridas, J.Miguel-Alonso, and Cruz Izu. Realistic Evaluation of Interconnection Network Performance at High Loads. *International Conference on Parallel and Distributed Computing Applications and Technologies*, 0:97–104, 2007.

[17] F.J. Ridruejo. INSEE: An Interconnection Network Simulation and Evaluation Environment. In *11th International Euro-Par Conference Proceedings, Lisbon, Portugal*, volume 3648 of *Lecture Notes in Computer Science*. Springer, 2005.

[18] TOP500 Supercomputer Sites. Lista de los 500 supercomputadores más potentes del mundo. `www.top500.org`, 2010.