A Strategy to Compute the InfiniBand Arbitration Tables *

F. J. Alfaro, José L. Sánchez
Dept. de Informática
Escuela Politécnica Superior
Universidad de Castilla-La Mancha
02071- Albacete, Spain
{falfaro, jsanchez}@info-ab.uclm.es

José Duato
Dept. de Informática de
Sistemas y Computadores
U. Politécnica de Valencia
46071- Valencia, Spain
jduato@gap.upv.es

Chita R. Das
Dept. of Computer Science
and Engineering
The Pennsylvania State Uni.
University Park, PA 16802
das@cse.psu.edu

Abstract

The InfiniBand Architecture (IBA) is a new industry-standard architecture for server I/O and interprocessor communication. InfiniBand is very likely to become the de facto standard in a few years. It is being developed by the InfiniBand^{SM} Trade Association (IBTA) to provide the levels of reliability, availability, performance, scalability, and quality of service (QoS) necessary for present and future server systems.

The provision of QoS in data communication networks is currently the focus of much discussion and research in industry and academia. IBA enables QoS support with some mechanisms. In this paper, we examine these mechanisms and describe a way to use them. We propose a traffic segregation strategy based on mean bandwidth requirements. Moreover, we propose a very effective strategy to compute the virtual lane arbitration tables for IBA switches. We evaluate our proposal with different network topologies. Performance results show that, with a correct treatment of each traffic class in the arbitration of the output port, every traffic class meets its QoS requirements.

1. Introduction

Input-output (I/O) buses have become the bottleneck for disk accesses, especially in high-performance servers. While buses have the major advantage of simplicity, and have served the industry well up to this point, bus-based I/O systems do not use their underlying electrical technology well enough to provide data transfer bandwidth out of a system to devices.

This was the primary reason why a group of the most important companies joined together to develop a standard for communication between processing nodes and I/O devices as well as for interprocessor communication, known as InfiniBand [6]. The InfiniBand^{SM} Trade Association (IBTA) is a group of more than 200 companies founded in August 1999 to develop IBA. Membership is also open to Universities, research laboratories, and others. The first specification of the InfiniBand Architecture (release 1.0) was published in October 2000 [5].

On the other hand, most of the current networking products have tried to achieve maximum throughput and minimum latency, leaving aside other aspects like guarantee of bandwidth, maximum latency deadline, interarrival delays, etc [9]. Many current applications need those characteristics that not all the current networks are able to provide. The Internet Engineering Task Force (IETF) is currently in the process of developing an architecture for providing QoS on the Internet. This effort is referred to as Differentiated Services [3].

In this paper, we propose a classification of the different traffic types with QoS needs, based on the proposal made in [7] (which was inspired by the Differentiated Services Architecture), as well as a strategy to compute the arbitration tables for the IBA switch ports to obtain the QoS required by the applications.

The structure of the paper is as follows: Section 2 presents a summary of the general aspects in the specifications of InfiniBand. In Section 3, we explain the most important mechanisms that InfiniBand provides to support QoS. In Section 4, we present our proposal, and its performance is evaluated in Section 5. Finally, some conclusions are given and future work is proposed.

2. InfiniBand

The InfiniBand Architecture (IBA) Specification describes a System Area Network (SAN) for connecting multiple independent processor platforms (i.e., host processor nodes), I/O platforms and I/O devices. The architecture is independent of the host operating system and processor platform.

IBA is designed around a switch-based interconnect technology with high-speed point-to-point links. An IBA

*This work was partly supported by the Spanish CICYT under Grant TIC2000-1151-C07

network is divided into subnets interconnected by routers, each subnet consisting of one or more switches, processing nodes and I/O devices. Routing in IBA subnets is distributed, based on forwarding tables stored in each switch. IBA supports any topology defined by the user, including irregular ones, in order to provide flexibility and incremental expansion capability.

IBA links are full-duplex point-to-point communication channels. The signaling rate on the links is 2.5 GHz in the 1.0 release, the later releases possibly being faster. Physical links may be used in parallel to achieve greater bandwidth. Currently, IBA defines three link bit rates. The lowest one is 2.5 Gbps and is referred to as 1x. Other link rates are 10 Gbps (referred to as 4x) and 30 Gbps (referred to as 12x) that correspond to 4-bit wide and 12-bit wide links, respectively. The width or widths that will be supported by a link is vendor-specific.

IBA switches route messages from their source to their destination based on forwarding tables that are programmed with forwarding information during initialization and network modification. Messages are segmented into packets for transmission on links and through switches. The packet size is such that after headers are considered, the Maximum Transfer Unit (MTU) of data may be 256 bytes, 1KB, 2KB or 4KB.

The IBA transport mechanisms provide several types of communication services between endnodes. These types are connections or datagrams, and both can be reliable (acknowledged) or unreliable. Obviously, for supporting the usual QoS requirements (guarantee of bandwidth, maximum latency deadline, interarrival delays, etc) applications must use reliable connections in order to be able to do resource allocation.

The interested reader is referred to the InfiniBand Specifications [5] for more details on InfiniBand. Other interesting papers that are good summaries of the official specifications are [8, 4].

3. IBA support for QoS

IBA provides three mechanisms that permit QoS to be supported: service levels, virtual lanes, and virtual lane arbitration for transmission over links. IBA defines a maximum of 16 service levels (SLs), but it does not specify what characteristics the traffic of each service level should have. Therefore, it depends on the implementation or the administrator how to distribute the different existing traffic types among the SLs. By allowing the traffic to be segregated by category, we will be able to distinguish between packets from different SLs and to give them a different treatment based on their needs.

IBA ports support virtual lanes (VLs), providing a mechanism for creating multiple virtual links within a single physical link. A VL represents a set of transmit and receive buffers in a port. IBA ports have to support a minimum of two and a maximum of 16 virtual lanes ($VL_0 \ldots VL_{15}$). All ports support VL_{15}, which is reserved exclusively for subnet management, and must always have priority over data traffic in the other VLs. Since systems can be constructed with switches supporting different numbers of VLs, the number of VLs used by a port is configured by the subnet manager. Also, packets are marked with a Service Level (SL), and a relation between SL and VL is established at the input of each link by means of a SLtoVLMappingTable. Each VL must be an independent resource for flow control purposes.

When more than two VLs are implemented, the priorities of the data lanes are defined by the VLArbitrationTable. This arbitration is only for data VLs, because VL_{15}, which transports control traffic, always has priority over any other VL. The VLArbitrationTable has two tables, one for scheduling packets from high priority VLs and another one for low priority VLs. However, IBA does not specify what is high and low priority. The arbitration tables implement weighted round-robin arbitration within each priority level. Up to 64 table entries are cycled through, each one specifying a VL and a weight, which is the number of units of 64 bytes to be transmitted from that VL. This weight must be in the range of 0 to 255, and is always rounded up as a whole packet.

A LimitOfHighPriority value specifies the maximum number of high priority packets that can be sent before a low priority packet is sent. More specifically, the VLs of the High_Priority table can transmit $LimitOfHighPriority \times 4096$ bytes before a packet from the Low_Priority table could be transmitted. If no high priority packets are ready for transmission at a given time, low priority packets can also be transmitted.

4. Our proposal

In this section we propose a traffic classification and a way of filling in the VLArbitrationTable of the output ports.

4.1. Traffic classification

In [7], a traffic classification was presented. The following four traffic categories were proposed: DBTS (Dedicated Bandwidth Time Sensitive), DB (Dedicated Bandwidth), BE (Best Effort) and CH (Challenged). We agree with this classification albeit with certain additions. For the DBTS and DB categories, we believe that a segregation of traffic should be made based on its characteristics. We have focused on the mean bandwidth requirements and have classified the connections into several classes according to their mean bandwidth requirements. By classifying connections into different categories, and by using appropriate values in the arbitration tables, we will be able to give a more adjusted treatment to each traffic class according to its needs. Specif-

ically, the categories that we distinguish for DBTS and DB are:

- Connections with very low bandwidth: connections with mean bandwidth requirements lower than or equal to 64 Kbps. An example of this class would be cellular phone conversations.
- Connections with low bandwidth: connections with mean bandwidth higher than 64 Kbps, but lower than or equal to 1.55 Mbps. An example of this class would be a video conference.
- Connections with high bandwidth: connections with mean bandwidth higher than 1.55 Mbps, but lower than or equal to 64 Mbps. An example of this class would be digital television transmission.
- Connections with very high bandwidth: connections with mean bandwidth higher than 64 Mbps. An example of this class would be high-definition television (HDTV) transmission.

InfiniBand distinguishes a maximum of 16 SLs and each port can have a maximum of 16 VLs. We can match the four previously established categories for DBTS and DB with InfiniBand SLs and assign a VL to each SL. Thus, we will use 4 VLs for the 4 DBTS SLs, another 4 VLs for the 4 DB SLs, one for BE traffic, one for CH traffic, and finally, one VL reserved by IBA for control traffic. Therefore, it would only be necessary for the ports to have 11 VLs. As 11 VLs are not allowed, 15 data VLs should be implemented. If the ports do not have enough VLs implemented for this classification, we can map several SLs to a single VL [2].

4.2. Filling in the virtual lane arbitration table

Pelissier [7] proposed to use the table of high priority for DBTS traffic and the table of low priority for the rest of the traffic, but he did not specify how to compute the weights for the VLs. In this section we are going to propose a strategy to fill in the weights for the arbitration tables. For the sake of clarity, we will assume that the network only has DB traffic and we will see how to fill in the table of low priority when the table of high priority is empty. In [2] we extend this proposal for DBTS traffic.

The control unit of each switch will be responsible for locally accepting or rejecting the connection requests based on the available local information. This information includes the state of the output links and how much bandwidth they have already reserved. Every time a switch accepts a connection, it will modify the arbitration table to adjust it to the traffic requirements. Alternatively these functions could be performed by the subnet manager if a centralized admission control is preferred.

Since we assumed that we do not have DBTS traffic, the table of high priority will be empty. As DB traffic requires guaranteed bandwidth, we can split time into frames of fixed duration. Each connection will be assigned a certain number of slots per frame, according to its bandwidth requirements.

According to the InfiniBand specifications, each arbitration table can have a maximum of 64 entries, each one with a maximum weight of 255. Assuming that each slot corresponds to the transmission of 64 bytes, which corresponds to a weight of one in a table entry (according to InfiniBand specifications), the maximum number of slots per frame is 64×255. We are going to compute the weights for each entry of the table with respect to this number. As we will see, this will drastically simplify the computation of table entries when connections are dynamically established.

Note that the important thing is neither the reference point nor the particular value of the weight for a certain table entry but the ratio between weights, which should agree with the ratio between the mean bandwidth requirements of the accepted connections. Therefore, we will compute the weight that should be assigned to a table entry according to the mean bandwidth of the connections serviced by that entry.

However, it should be noted that the InfiniBand arbitration scheme will assign all the bandwidth to the set of established connections, even if these connections requested a lower overall bandwidth. The reason is that once all the virtual channels in the corresponding table of priority have been serviced, the link arbiter will repeat the cycle again. Therefore, all the link bandwidth will be consumed (assuming that there are packets ready to be transmitted) regardless of the values in the tables of priority. Thus, in order to simplify the computation of the values in the tables of priority, we decided to compute these values as if all the slots were in use. By doing so, the entries in the tables do not need to be recomputed when a new connection is established because all of them have been computed with respect to the same reference. Note that each connection will receive more bandwidth than requested unless the full link bandwidth has been reserved. But this will happen anyway, regardless of how the values in the entries are computed. Therefore, our methodology has the significant advantage of not requiring recomputing the table entries for previously established connections when a new connection is established, achieving the QoS guarantees.

As we only have 64 entries in each arbitration table, this could limit the number of connections that can be accepted. A connection requiring very high bandwidth could also need slots in more than one table entry. For this reason, we propose grouping the connections with the same SL into a single table entry until completing the maximum weight for that entry, then moving to another free entry. This way,

the number of table entries is not a limitation for the acceptance of new connections, but only the available bandwidth.

The weight must therefore be computed according to the total mean bandwidth requested by the connections serviced by the corresponding table entry, and every time a new connection is accepted, only the table entries used by this connection need to be recomputed. Let us now compute the weight corresponding to a certain table entry.

Each frame consists of $64 \times 255 = 16320$ slots of 64 bytes. For the different link rates of InfiniBand we can compute its T_{frame}, that is, the time it would take to transmit a frame assuming that all the table entries have been assigned their maximum value. The values of this time for each link rate are: 3.342336 msec for 1x (2.5 Gbps), 0.835584 msec for 4x (10 Gbps) and 0.278528 msec for 12x (30 Gbps).

The bandwidth B assigned to a connection is the number of slots assigned to that connection divided by the total number of slots in the frame and multiplied by the link bandwidth, that is, the percentage of the frame assigned to that connection multiplied by the link bandwidth.

$$B = \frac{N_{Slots}}{N_{Slots\ per\ frame}} \times B_{link}$$

So, the number of slots for a connection with mean bandwidth B bps is

$$N_{Slots} = \left\lceil \frac{B}{B_{link}} \times 64 \times 255 \right\rceil$$

where $\lceil\ \rceil$ is to round up. We can see T_{Frame} as

$$T_{Frame} = \frac{64\ entries \times 255\ \frac{units}{entry} \times 64\ \frac{bytes}{unit} \times 8\ \frac{bits}{byte}}{B_{link}}$$

This way, we can compute the number of slots as

$$N_{Slots} = \left\lceil B \times \frac{T_{frame}}{512} \right\rceil$$

Finally, the value assigned to a table entry is the sum of slots for all the connections assigned to that entry.

In order to guarantee that some bandwidth will be available for traffic with no QoS requirements (BE and CH), it is possible to reserve some slots for these traffic classes. The amount of reserved bandwidth will depend on system requirements. Therefore, when the network starts to establish DB connections, the entries corresponding to BE and CH traffic will have already been created in the table of low priority. Note that when no other traffic is available, if there are BE or CH packets, they will be selected for transmission. The values in the arbitration table for BE and CH traffic guarantee a minimum bandwidth, but do not limit the bandwidth available to these categories when no other traffic is available. This is a good feature of the weighted round robin arbitration policy.

5. Performance evaluation

In this section, we evaluate the behavior of our proposal. Instead of analytic modeling, simulation was used. We have developed a flit-level simulator that models the InfiniBand behavior.

We have used both regular and irregular networks. Regular networks have been chosen with hypercube and mesh topology. Irregular ones have been randomly generated. In all the cases, every switch has four ports for interconnection between switches and other 4 ports with one host attached to each of them.

To be able to test our proposal, each port has 15 data VLs both on the switches and on the hosts. Although we only need 11 VLs, this is not a valid value in InfiniBand for the number of VLs for a port. Data VLs will be used according to the traffic classification seen in Section 4.1. Each VL has its own buffer, both at the input and the output of a switch. The size of these buffers has been assumed to be proportional to the packet sizes we have tried: 256 and 4096 bytes. For these packet sizes, the buffer sizes we have used are 1024 and 16384 bytes, respectively. Therefore, these buffer sizes allow 4 entire packets to be stored.

With respect to network size, we have evaluated networks with sizes ranging from 8 to 64 switches (with 32 to 256 hosts, respectively). We have tried the three link rates of InfiniBand. Due to space limitation, and taking into account that results are similar, we will only include here the results for the networks with 16 switches for a link rate of 2.5 Gbps. The other results are available in [1].

The crossbar incorporated in the switches is multiplexed among VLs, having the same number of inputs as input ports in the switch. Therefore, for the networks analyzed in this paper, with 8-port switches, all the switches have crossbars of size 8×8. To prevent the crossbar from becoming a bottleneck, the crossbar ports work twice as fast as the links so that they are able to take out everything that arrives [10].

With respect to traffic model, we have used CBR traffic of type DB. Although the results for BE and CH traffic are not the focus of this paper, we have reserved slots for these traffic classes in the tables. In this first approach, we have not considered DBTS traffic, but in [2] we present results for DBTS traffic when several mixtures of DBTS and DB traffic are used. Those results show that both kinds of traffic are able to achieve their expected performance.

CBR traffic is randomly generated among the 4 SLs considered in Section 4.1, and requested bandwidth is uniformly distributed across the range for each SL. Specifically, for the four categories defined there, we will establish connections with the following ranges: SL 0 (8 – 64 Kbps), SL 1 (64 Kbps – 1.55 Mbps), SL 2 (64 Kbps – 64 Mbps) and SL 3 (64 – 300 Mbps). We can observe that the bandwidth range for each SL is very wide, and therefore, packets with very different characteristics are going to coexist in the

	Topology					
	Irregular		Hypercube		Mesh	
	Small	Large	Small	Large	Small	Large
Injected traffic (Bytes/Cycle/Node)	0.7691	0.7571	0.7675	0.7556	0.7516	0.7582
Delivered traffic (Bytes/Cycle/Node)	0.7691	0.7571	0.7675	0.7556	0.7516	0.7582
Average utilization for host interfaces (%)	79.32	75.86	79.15	75.71	77.51	75.97
Average utilization for switch ports (%)	79.90	77.50	79.85	77.92	79.68	77.76
Average reservation for host interfaces (Mbps)	1920.69	1892.67	1916.68	1889.04	1876.87	1895.45
Average reservation for switch ports (Mbps)	1938.41	1933.69	1958.08	1944.06	1929.58	1940.23

Table 1. Traffic and utilization for different topologies and packet sizes.

same buffers and in the same SL.

The workload in the network is established by randomly trying a number of connections for each SL higher than the connections that can be accepted. In this way, these connections are equally distributed among the different SLs defined. When the maximum number of connections have been established, the connection establishment phase finishes and a transient period begins. This transient period lasts until 10000 packets have arrived at their destinations. Once the transient period finishes, the steady state period begins, where we will gather the data to be shown in the next section. The steady state period continues until the connection with a smaller mean bandwidth has received 100 packets.

5.1. Simulation results

In this section, we show the performance evaluation results. For simulations, we have reserved 20% of link bandwidth for BE traffic. So, for link speed of 2.5 Gbps, the maximum bandwidth available for CBR traffic is 2 Gbps. For each topology tested, we can see in Table 1 the injected and delivered traffic (in bytes/cycle/node), the average utilization (in %) and the average bandwidth reserved (in Mbps) in host interfaces and switch ports.

Note that the behavior is similar for all the networks and for both packet sizes. Moreover, with our proposal, the network can reach a throughput close to 80%, which is the maximum available for this kind of traffic, because the other 20% is reserved for BE traffic. The reason is that only accepted connections are allowed to transmit packets. Also, note that for small packet size the network reaches a slightly higher throughput. This is due to the fact that the overhead introduced by packet headers is more important for small packet size and more packets must be sent. Note that connections have been accepted until the payload approaches 80%. As all kind of networks and both packet sizes have a similar behavior, in the rest of this paper only results for the irregular network with small packet size are going to be shown. The other results are available in [1].

In order to get more information about the distribution of packet delay, the percentage of packets whose delays are lower than a set of thresholds have been computed. These results give an idea of the percentage of packets that will meet a given deadline, and therefore, they measure the provided QoS. The thresholds are different for each type of connection, and they are related to their inter-arrival time (IAT). Therefore, thresholds become tighter as bandwidth requirements increase. The results are presented in Figure 1 for each SL. The packets for very low and low bandwidth connections (SL 0 and SL 1) arrive before their $\frac{IAT}{32}$. For high and very high bandwidth connections (SL 2 and SL 3), this value is too small and packets need some more time, but in both cases all packets have arrived before a deadline equal to $\frac{3 \times IAT}{4}$. These are very good results that show that our methodology to compute InfiniBand arbitration tables is very effective and provides good QoS support.

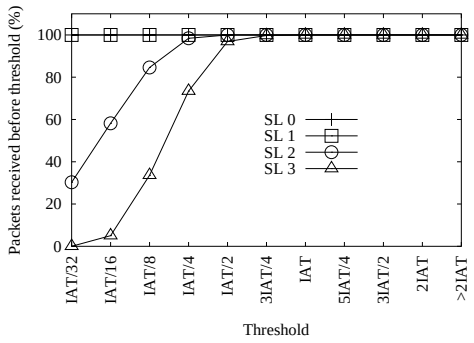

Figure 1. Distribution of packet delay.

Average packet jitter has also been measured. We have established several intervals to compute the jitter and bar graphs plot the percentage of packets received within each interval. These intervals are different for each type of connection, and are related to their IAT. The results are shown in Figure 2. We can observe that packets from SL 0 and SL 1 always arrive with jitter almost equal to zero (in the interval $[\frac{-IAT}{8}, \frac{IAT}{8}]$). For packets from SL 2 and SL 3 the jitter has a Gaussian distribution never exceeding $\pm IAT$.

Finally, we have studied the influence of mixing connections with very different mean bandwidth on the same service level (and therefore, on the same virtual lane). Given a deadline, we have selected the connections that have delivered the lowest and the highest percentage of packets before that deadline. We have selected a very tight deadline so that the percentage of packets meeting the deadline was lower than 100%. In particular we have selected a dead-

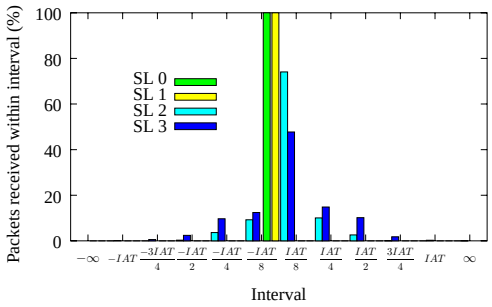

Figure 2. Average packet jitter.

line equal to $\frac{IAT}{2}$. In the figures these connections will be referred to as the worst and the best connections, respectively. All packets for SL 0 and SL 1 have arrived before their first threshold because for these connections the IAT is long enough. As the results do not vary with packet size, and due to lack of space, we have only shown in Figure 3 results for SL 2 and SL 3 for irregular network with small packet size.

The results show a similar behavior for all the connections within the same service level. Note that even the worst connections for each service level have a deadline lower than their IAT. Therefore, our methodology to compute the weights for the table of priority is very robust, providing good QoS guarantees to all the connections, despite the fact that connections with very different bandwidth requirements share the same resources (virtual lanes).

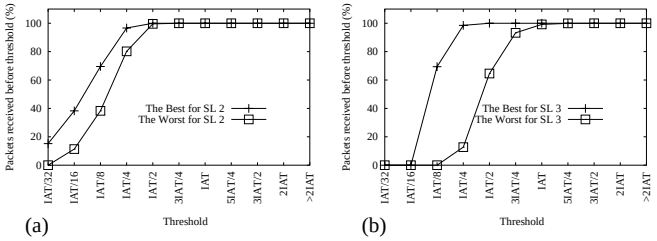

Figure 3. Behavior of the best and the worst connection. (a) SL 2 and (b) SL 3.

6. Conclusions

We have proposed a methodology to compute the virtual lane arbitration tables for InfiniBand. Our proposal has the significant advantage of not requiring recomputing the table entries for previously established connections when a new connection is established. This is because all the entries have been computed with respect to the same reference.

We have tested the behavior of the proposed methodology by simulation using CBR traffic with different mean bandwidth requirements. We have seen that the maximum utilization that can be reached is almost the maximum available for both small and large packet sizes, regardless of the network topology. We have also shown that all types of traffic can satisfy their QoS requirements. In particular, we have evaluated the percentage of packets that arrive before a certain deadline. Even for service levels (SLs) with the lowest interarrival time (IAT), almost all the packets arrive before their $\frac{IAT}{2}$ or soon after. We have also shown that all the connections sharing the same SL and virtual lane (VL) have very similar behavior, despite the fact they have very different mean bandwidth requirements. In particular, we have seen that the best and the worst connection have a similar behavior.

We have also evaluated jitter, showing that all packets arrive with jitter between $\frac{-IAT}{8}$ and $\frac{IAT}{8}$ for connections with low bandwidth requirements. For connections with high bandwidth requirements the jitter has a Gaussian distribution with all the packets within the interval $[-IAT, IAT]$.

In [2] we have extended and tested our proposal for traffic that has a latency deadline (DBTS). We are currently examining a number of extensions to this work. We want to test this proposal with VBR traffic. Second, we are studying different classifications for the case when we have fewer VLs than SLs. Third, we are exploring other alternative designs with the goal to improving the QoS support. Finally, we would like to test our proposal in an InfiniBand commercial product as soon as there is one available, in order to verify the practicality of our model.

References

[1] F. Alfaro, J. Sánchez, and J. Duato. A Strategy to Compute the InfiniBand Arbitration Tables. Technical Report DIAB-01-02-20, Departamento de Informática Universidad de Castilla-La Mancha, Oct. 2001. `http://raap.info-ab.uclm.es/diab-01-02-20.pdf`.

[2] F. J. Alfaro, J. L. Sánchez, and J. Duato. A Strategy to Manage Time Sensitive Traffic in InfiniBand. In *Proceedings of Workshop on Communication Architecture for Clusters (CAC)*, Apr. 2002. Held in conjunction with IPDPS'02, Fort Lauderdale, Florida.

[3] S. Blake, D. Back, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. *RFC 2475*, Dec. 1998.

[4] P. by ISSG Technology Communications. Infiniband architectural technology. Technical Report TC000702TB, Compaq Computer Corporation, July 2000.

[5] InfiniBand Trade Association. *InfiniBand Architecture Specification Volume 1. Release 1.0*, Oct. 2000.

[6] $InfiniBand^{TM}\ Trade\ Association$. *http://infinibandta.com*.

[7] J. Pelissier. Providing quality of service over Infiniband architecture fabrics. In *Proceedings of the 8th Symposium on Hot Interconnects*, Aug. 2000.

[8] G. Pfister. *High Performance Mass Storage and Parallel I/O*, chapter 42: An Introduction to the InfiniBand Architecture, pages 617–632. IEEE Press and Wiley Press, 2001.

[9] M. Schwartz and D. Beaumount. Quality of service requirements for audio-visual multimedia services. *ATM Forum*, ATM94-0640, July 1994.

[10] A. Systems. Avici Terabit Switch Router. http://www.avici.com/.