University of Castilla-La Mancha

A publication of the **Department of Computer Science**

Providing QoS over Advanced Switching

by

Raúl Martínez, Francisco J. Alfaro, José L. Sánchez, Tor Skeie

Technical Report **#DIAB-05-01-5** January 2005

This work has been submitted to the
International Conference on Parallel Processing 2005 (ICPP-2005)

DEPARTAMENTO DE INFORMÁTICA
ESCUELA POLITÉCNICA SUPERIOR
UNIVERSIDAD DE CASTILLA-LA MANCHA
Campus Universitario s/n
Albacete - 02071 - Spain
Phone +34.967.599200, Fax +34.967.599224

Contents

Providing QoS over Advanced Switching[1]

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

[1]This work was partly supported by the Spanish CICYT under Grant TIC2003-08154-C06.

Abstract

The Advanced Switching Interconnect (ASI) is a new open standard fabric-interconnect technology for communications, storage and embedded environments, which provides the advanced features of existing proprietary fabrics in an open standard developed by the ASI Special Interest Group.

The provision of quality of service (QoS) in computing and communication environments is currently the focus of much discussion and research in industry and academia. ASI provides some mechanisms that correctly used permit us to provide QoS. Specifically, we use virtual channels, traffic classes, the egress link scheduling, and the admission control mechanism. In this paper, we examine these mechanisms and explain both how to provide a differentiated treatment establishing a priority system, and how to provide QoS guarantee. Specifically, we are going to provide QoS based on minimum bandwidth and maximum latency requirements. A special discussion is given over to the possible implementations of the egress link scheduling of the AS elements to obtain the QoS required by the applications.

1 Introduction

The PCI bus has served industry well for the last 10 years and is currently used extensively in a wide variety of personal and enterprise computing systems. However, the processors and I/O devices of today and tomorrow are demanding much higher I/O bandwidth than PCI 2.2 or PCI-X can deliver. The reason for this limited bandwidth is the parallel bus implementation. PCI Express [9] eliminates the legacy shared bus-based architecture of PCI and introduces an improved and dedicated point-to-point full-duplex interconnect. However, key PCI attributes, such as its usage model and software interfaces, are maintained. PCI Express has been adopted by the nearly 1000 members of the PCI-SIG [8] and is the dominant host-centric interconnect architecture.

The Advanced Switching Interconnect Special Interest Group (ASI SIG) [7] is a non-profit collaborative trade organization tasked with developing and supporting a switched interconnect and data fabric interface specification for communications, storage and embedded systems, termed Advanced Switching Interconnect (ASI) or just Advanced Switching (AS), based on the PCI Express architecture. In the final days of 2003, the ASI SIG published v1.0 of the Advanced Switching specifications [1].

Advanced Switching is built on the same physical and link layers of PCI Express technology. Moreover, it includes an optimized transaction layer to enable essential communication capabilities, including protocol encapsulation, peer-to-peer communications, some mechanisms that permit us to provide quality of service (QoS), enhanced fail-over, high availability, multicast communications, and congestion and system management. Some target applications for AS include: Bladed computing, embedded computers (e.g. medical imaging, military command and control), communications edge/access equipment, communications routers, enterprise and high performance storage routers and arrays.

The provision of QoS in computing and communication environments is currently the focus of much discussion and research in industry and academia. For example, the Internet Engineering Task Force (IETF) has two proposals for providing QoS on the Internet. The first one is referred to as Differentiated Services [4]. This approach distinguishes between different kinds of traffic and provides different treatment for these categories. The second one is referred to as Integrated Services [5]. The latter provides QoS guarantees for the applications based on their requirements requested during the connection establishment phase.

AS provides mechanisms that permit QoS to be supported. Specifically, an AS fabric permits us to employ virtual channels, traffic classes, egress link scheduling and an admission control mechanism to provide a different treatment to the traffic of the different service classes. In this paper, we will explain both how to use the Advanced Switching mechanisms to provide a differentiated treatment establishing a priority system, and how to provide QoS guarantee. Specifically, we are going to provide QoS based on minimum bandwidth and maximum latency requirements. A special discussion is given over to the possible implementations of the egress link scheduling of the AS elements 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 ASI. In Section 3, we explain the most important mechanisms that ASI provides to support QoS. In Section 4, we present our proposal to use the above mechanisms, and some aspects are evaluated in Section 5. Finally, some conclusions are given and future work is proposed.

2 Advanced Switching

Advanced Switching (AS) is a multipoint, peer-to-peer switched fabric architecture designed to provide the functionality of the proprietary interconnects that have been at the core of storage, communications, and embedded computing systems in an open standard. The scalable and extensible AS fabric architecture supports high availability capabilities such as hot add/remove, redundant pathways, and fabric management failover.

The AS architecture is built upon the data link and physical layers established by the PCI Express architecture to achieve widespread interoperability and cost-effective reuse of technology. The physical layer consists of a dual-simplex channel that is implemented as a transmit pair and a receive pair. A data clock is embedded using the 8b/10b encoding scheme. The initial frequency is 2.5 Gbps in each direction. The bandwidth of a link may be linearly scaled by adding signal pairs to form multiple lanes. The physical layer supports x1, x2, x4, x8, x16, or x32 lane widths. The link layer is responsible for data integrity and adds a sequence number and a CRC to the transaction layer. The link layer will automatically retry a packet that was signaled as corrupt. A credit-based flow control protocol ensures that packets are only transmitted when the buffer at the other end is able to receive those packets. Over the physical and link layers of PCI Express, AS implements an optimized transaction layer, providing a rich set of features and capabilities.

For unicast traffic the AS transaction layer provides source-based path routing versus the memory mapped routing of PCI Express. By eliminating the hierarchical structure of memory mapped routing, flexible topologies can be constructed such as star, dual-star, and full mesh.

AS also supports multicast traffic. Multicast routing enables a single packet generated by a source to be sent to multiple endpoints. Duplicate packets are generated at points along the fabric where the associated multicast distribution tree branches.

AS encapsulates data packets and attaches a header that routes it through the fabric regardless of the packet format. The header contains a Protocol Interface (PI) field that is used at the packet destination to determine the packet's format. Thus, nearly any transport, network, or link layer protocol can be routed through an AS network. PCI Express packets are a particularly important format for system developers where AS will serve as a central switch fabric connecting PCI Express endpoints. PI-8 allows multiple-enabled PCI Express CPUs to connect transparently to multiple-enabled PCI Express I/O nodes through the AS fabric using PCI Express plug-and-play software.

In addition to the link layer credit-based flow control mechanism, AS defines other mechanisms for dealing with AS fabric congestion, for example: Status-based flow control, packet dropping and endpoint injection rate limiting. An AS fabric may utilize management software capable of implementing additional congestion management mechanisms such as path selection, admission control, or dynamic adaptation of switch element behavior.

3 AS Support for QoS

AS provides mechanisms that permit QoS to be supported. Some of these mechanisms provide the AS hardware with the ability to support high throughput traffic with low and predictable latency. Specifically, an AS fabric permits us to employ virtual channels, traffic classes, egress link scheduling and an admission control mechanism to provide a different treatment to the traffic of the different service classes. The goal is to move efficiently traffic classified into differentiated service classes and to avoid congestion problems within one or more of these classes, even when traffic volume approaches the AS fabric capacity.

3.1 Virtual Channels

Virtual Channels (VCs) provide a means of supporting multiple independent logical data flows over a given common physical channel. Conceptually, this involves multiplexing different data flows onto a single physical link. AS supports up to 20 VCs of three different types: Up to 8 bypass capable unicast VCs (BVCs), up to 8 ordered-only unicast VCs (OVCs), and up to 4 multicast VCs (MVCs). The AS architecture defines the automatic hardware negotiation process for enabling, on a per link basis, the largest common number of VCs of each type.

The BVCs are unicast VCs with bypass capability, necessary for deadlock free tunneling of some protocols (typically load/store ones). This mechanism works in the following way: When a packet arrives at the VC it is stored in a FIFO queue. Once a packet reaches the head of this queue it is transmitted if there are flow control credits. If a bypassable packet is at the head of that queue but there are no flow control credits, it is moved to another queue where it waits until there are flow control credits available. OVCs are FIFO queue unicast VCs. Two single bit flags, the ordered-only flag and the type-specific flag, within an AS header specify the required packet VC type and whether it is bypassable.

3.2 Traffic Classes

The AS architecture enables multiple upper level protocols, each one possibly supporting many separate flows and multiple service classes, to be tunneled through an AS fabric. The characteristics and requirements of each traffic class (TC) are defined to

support one or more upper level protocols. This model enables AS ingress interfaces to schedule individual AS flows via the TC parameters specified for each flow.

The packet's TC identifier is transmitted unmodified from source to destination through an AS fabric. At each hop within an AS fabric, the TC identifier contained in the packet's AS route header is used to apply appropriate VC selection. Each VC type is governed by a separate and distinct TC/VC mapping. The TC field in the AS packet header is a 3-bit field specifying one of eight possible TCs.

AS ingress interfaces must map TCs to the VC queues required for the AS transaction layer. Since systems can be constructed with switches supporting a different number of VCs, TC to VC mappings can change in each hop through a switch fabric. Thus, VCs themselves may be aggregated (when the next hop switch implements fewer VCs) and disaggregated (when the next hop switch implements more VCs).

3.3 Egress Link Scheduling

AS includes a scheduler to resolve between the up to twenty VCs competing for bandwidth onto the egress link. AS defines two egress link schedulers: The VC arbitration table scheduler and the minimum bandwidth (MinBW) egress link scheduler. A given implementation may choose any of them or may implement its own proprietary mechanism.

3.3.1 Virtual Channel Arbitration Table Scheduler

The VC arbitration table scheduler is similar to the one defined by PCI Express. It provides a packet-based weighted round robin arbitration between the VCs, employing a VC arbitration table to schedule the next packet to be transmitted.

The VC arbitration table is a register array with fixed-size entries of 8 bits. Each 8-bit table entry corresponds to a slot of a WRR arbitration period. Each entry contains a field of 5 bits with a VC identifier value and a reserved field of 3 bits. The VC identifier indicates that the corresponding slot within the WRR arbitration period is assigned to the VC indicated by the VC identifier. The table is cycled through, and if the VC of an entry has no packet ready to be transmitted, that entry is skipped. The number of entries of the VC arbitration table may be 32, 64, 128, 256, 512, or 1024.

3.3.2 Minimum Bandwidth Egress Link Scheduler

The MinBW egress link scheduler, or just MinBW scheduler, is intended for a precise allocation of bandwidth. Figure 1 shows the organization of the MinBW scheduler, which consists of two parts. The first one, the MinBW scheduler, is a mechanism to provide a single VC, the Fabric Management Channel (FMC), with absolute highest priority servicing, ahead of the other VCs, but with its bandwidth limited by a token bucket. The second one is a mechanism to distribute bandwidth amongst the rest of the VCs according to a configured specification of the relative bandwidth to be allocated

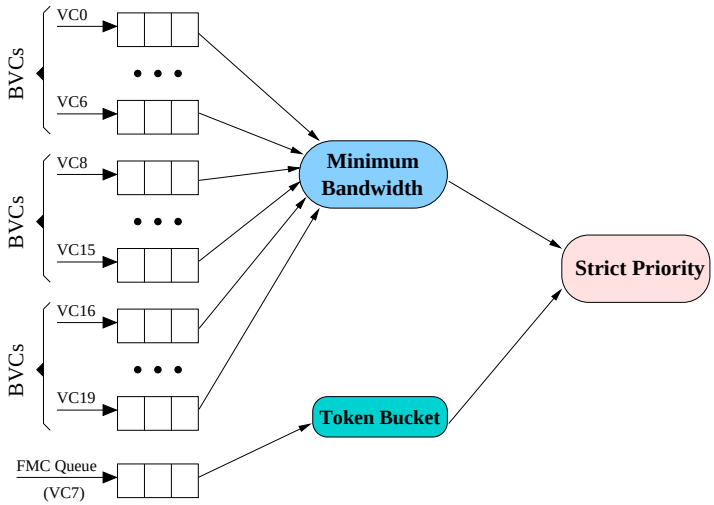


Figure 1: Minimum Bandwidth Egress Link Scheduler for a port with 20 VCs.

to each of those VCs. This mechanism allows us to guarantee a minimum bandwidth to each of the VCs.

AS does not specify an algorithm or implementation for the MinBW scheduler, but it must respect certain properties:

- **Work conserving:** If at least one VC has a packet available to be sent over the egress link, then it should be transmitted. This statement assumes that link credit exists to move packets.

- **Bandwidth metering, not packet metering:** The MinBW scheduler allocates link bandwidth to each VC, taking account of packet sizes as it chooses VCs. If packets are of variable size, VCs holding small packets must be selected more often than other VCs holding large packets, assuming of course that each VC has been allocated equal minimum bandwidth.

- **Minimum bandwidth guarantee:** Egress link bandwidth is allocated among the VCs in proportion to a set of configurable weights that represent the fraction of egress link bandwidth assigned to each VC. Generally, the weights should total 100% of the egress link bandwidth. However, the weights are relative and therefore the total need not be 100%.

- **Fair redistribution of unused bandwidth:** Bandwidth left over after all the VCs have consumed their configured minimum bandwidth, must be redistributed among those VCs that have packets to transmit and credits to do so in proportion to their bandwidth allocations.

- **Memoryless:** During the time that a VC is not allowed to transmit any packet due to the credit-based flow control it does not consume bandwidth and the scheduler must not save that VC's MinBW allocation for future use. Hence, if a VC receives enough credits to transmit when it previously has had none, it is

only allotted its MinBW and its fair share of the excess bandwidth. Thus, it does not receive the opportunities missed when it was not allowed to transmit due to lack of credits.

3.4 Admission Control

Moreover, fabric management software may regulate access to the AS fabric, allowing new packet flows entry to the fabric only when sufficient resources are available. Fabric management software may track resource availability by monitoring AS fabric congestion and tracking active packet flows and their bandwidth. This is very useful when traffic flows are predominately connection-oriented and carefully rate-limited. In an implementation employing admission control, sources would only add new flows when it is permitted by the fabric management admission control module. Such software allows a new flow access to the AS fabric only if it cannot create congestion. If it is admitted, the software assigns a suitable bandwidth, traffic class and path to the traffic. As necessary, the software may reduce the bandwidth assigned to existing flows, or even terminate an existing one, to accommodate a new flow. AS specifications just cite admission control as a possible mechanism to be used, but do not give any indication of how to implement it.

4 Our Proposal

In this section, we propose a way of using some of the above presented AS mechanisms in order to provide QoS in this interconnection technology. We are going to use the traffic classification presented in [10], which is based on traffic requirements. Specifically, we use the following traffic types:

- DBTS: Dedicated Bandwidth Time Sensitive traffic, which requires a given minimum bandwidth and must be delivered within a given latency in order for the data to be useful. Examples of such data streams include video conference and interactive audio.
- DB: Dedicated Bandwidth traffic, which requires a given minimum bandwidth but is not particularly sensitive to latency. This includes traffic that must have a bounded latency but where the actual value of the bound is not critical. An example of this class of traffic could be the playback of a video clip.
- BE: Best-Effort. This traffic tends to be bursty in nature and largely insensitive to both bandwidth and latency. The majority of traffic in current networks is of this type, like file and printing services or web browsing.

AS specifications allow a link to be multiplexed into up to 16 unicast VCs. Note that combining the 8 available TCs and the two possible unicast VC types, we obtain 16 different combinations. We call each of them a Service Class and we propose to

define up to 16 service classes and assign each of them to a different Traffic Class and unicast VC type pair. The bypassable TC 7 would be assigned to the network control traffic. If there are enough VCs we will devote a different VC to each existing service class. However, if there are not enough implemented VCs along the whole path of a connection, more than one service class should be assigned to the same VC.

4.1 Connection Admission Control

To provide QoS guarantees a connection admission control (CAC) must be used. If a CAC is not used it is only possible to obtain a scheme of priorities where some service classes would have a higher priority than others, but no guarantee could be provided.

The CAC monitors the bandwidth that each connection with QoS requirements has reserved all along its path. Moreover, it rejects a new connection if there are not enough resources to accommodate the new connection without affecting the previously established ones. In this approach, the best-effort traffic would have assigned a minimum bandwidth but no admission control would be implemented over this traffic.

4.2 Egress Link Scheduling

As was stated, AS includes a scheduler to resolve between the output VCs. Therefore, in addition to the traffic segregation according to the application's characteristics, the scheduler must process appropriately each service class to provide it with its QoS requirements. Note that if we want to provide applications with latency guarantee the only specified mechanism that we can use is the VC arbitration table because, as was stated, the MinBW egress scheduler is only able to distribute bandwidth, but it cannot deal with latency requirements.

Therefore, we propose using the VC arbitration table scheduler to provide bandwidth and latency guarantee. Obviously, the VC arbitration table entries must be properly distributed among the different VCs that accommodate the defined service classes.

In [3] we explain how to configure this kind of arbitration table (in the case in question for InfiniBand) to provide bandwidth and latency guarantee. In order to provide traffic of a given VC with a minimum bandwidth, the number of entries assigned to that VC must be proportional to the desired bandwidth. For example, if we want to reserve 25% of the egress link bandwidth to a VC, then 25% of the entries of the table must be assigned to that VC. To provide maximum latency requirements to the traffic of a VC, the maximum time must be studied that a packet can spend crossing a network element as well as the time it takes in being transmitted to the next element once it has been chosen by the scheduler. With this information it is possible to control the maximum latency of a network element crossing, by fixing the maximum separation between two consecutive entries devoted to that VC.

This way of assigning the entries of the table faces the problem of bonding the bandwidth and latency assignments. If one sets a maximum separation between two

consecutive entries of a VC, one is indirectly assigning a certain number of table entries and so a minimum bandwidth to that VC. This can be a problem because the most latency-restrictive traffic does not usually require a high bandwidth reservation. However, our view is that this problem can be minimized by using accurately the CAC mechanism.

Another major problem of the AS arbitration table is that it does not operate properly with variable packet sizes. If two VCs have the same number of entries assigned, but the average packet size of their respective traffic flows is not the same, they are not actually getting the same bandwidth. The problem is settled in this paper by using a fixed packet size. We are, however, studying different ways of using the table in order to be able to use a variable packet size.

Although they are no compulsory for the AS VC arbitration table, note that the operation system of this table scheduler presents all the required properties of the MinBW scheduler except the *bandwidth metering*, because the way the table works is based on the number of packets transmitted instead of just bandwidth transmitted.

There are two possible ways of using the table. The first possibility is to fill in the table in advance, defining a set of service classes with a different minimum bandwidth and maximum latency reservation [11]. This distribution would be made taking into account the expected use of each service class. The different connections would be assigned to a service class attending to their type. The second possibility consists in distributing the entries according to the connection requirements in a dynamic way. In this approach the table may be modified both when a new connection is accepted and when a previously established connection ends [2]. This allows more flexibility and a more accurate use of the resources. Note that the second possibility is the appropriate approach to provide QoS guarantee while the first one is more appropriate for a priority system approach.

Summing up, our proposal to provide any kind of guarantee consists in using the CAC to manage correctly the available resources. Moreover, if we want to provide applications with bandwidth and latency guarantee we propose to use the AS VC arbitration table scheduler, and to assign dynamically the arbitration table entries to the VCs according to the requirements that the traffic flows using those VCs have. However, if we only want to provide applications with bandwidth guarantee (no latency guarantee) or to provide a differentiated treatment to the different service classes, we can also use the MinBW scheduler.

5 Performance Evaluation

In this section we evaluate the behavior of our proposals. For this purpose, we have developed a detailed simulator that allows us to model the network at the register transfer level following the AS specifications. First, we will describe the main AS network model features and the traffic models that we have used. Secondly, the configuration and implementation of the egress link scheduler are specified, and finally we present

and analyze the obtained results.

5.1 Network Model

Two different network topologies have been used in the simulations. The first, shown in Figure 2, consists of three hosts interconnected by a single switch and is intended to allow us to analyze the egress link scheduler performance with the minimum influence possible of other factors. The second topology consists of 32 hosts connected using a 32x32 multistage network and is used for studying the performance results in a more realistic topology. In the latter, more complex interactions take place and other factors, such as congestion, can appear. This network has switches with 8 ports. The width lane is x1, so the link bandwidth is 2.5 Gbps, but with the 8b/10b encoding scheme the maximum bandwidth for data traffic is only 2 Gbps.

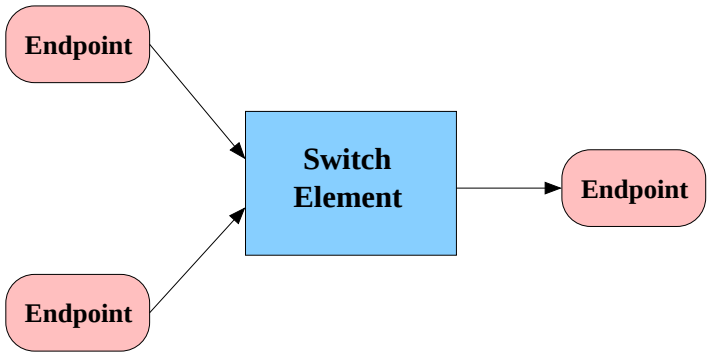


Figure 2: Single switch network model.

The switch model is shown in Figure 3. It has buffers both at the input and at the output, and a crossbar as commutation element. Virtual output queues (VOQs) are used at the input. Each port has 8 BVCs and each VC has a number of FIFO queues equal to the number of switch ports, although the queues of the same VC share the same credit count. To prevent the crossbar from becoming a bottleneck, it works twice as fast as the network links. Due to the VOQs the routing units are prior to the input buffers.

5.2 Traffic Model

In this paper we have considered 7 service classes (SCs), 2 for best-effort traffic, 2 for DB traffic and the other 3 for DBTS traffic. We have also considered another service class for network control traffic but we have not injected it on simulations. We have used the BVCs but all packets have their bypassable bit set to false, so the results are the same as using OVCs. We have injected CBR traffic of different bandwidth for each one of the 7 service classes considered on the simulations. In all cases, packet size is 512 bytes. Each figure shown in this paper has been obtained running 40 simulations. On each simulation the traffic injected by SC2 to SC6 is the same, but SC0 and SC1 increase the amount of traffic injected in each simulation in order that the network can

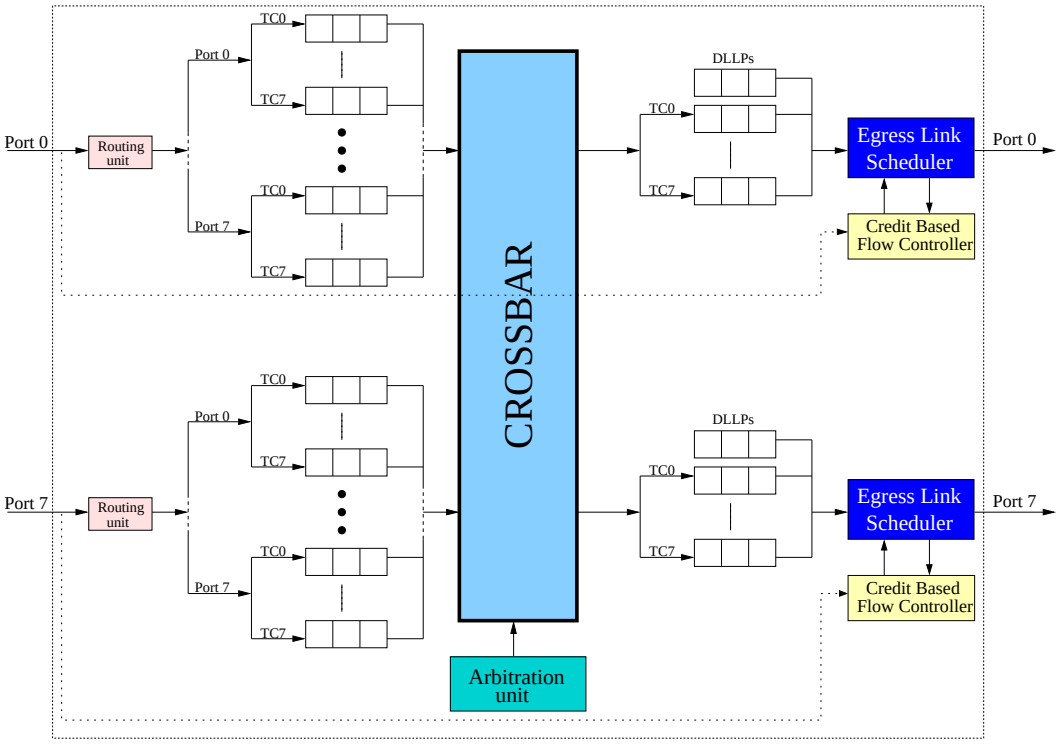


Figure 3: Switch model.

achieve a saturation status. The traffic generated per service class is specified in Table 1. In the case of the single switch network all the traffic is addressed to the same host, but in the multistage network the traffic is uniformly distributed among all the hosts. Taking into account the traffic generated by all the hosts, the total injection rate of each service class is shown in Figure 4.

Table 1: Traffic injected per host and the VC arbitration table configuration.

		Injected traffic (Gb/s)		**Table configuration**		
SC	**Type**	**Single**	**Multistage**	**# entries**	**% entries**	**Max. sep.**
7	Control	-	-	-	-	-
6	DBTS	0.1	0.2	8	25	3
5	DBTS	0.15625	0.3125	5	15.625	6
4	DBTS	0.15	0.3	4	12.5	7
3	DB	0.15625	0.3125	5	15.625	-
2	DB	0.25	0.5	5	15.625	-
1	Best Effort	0.001-0.15	0.002-0.3	3	9.375	-
0	Best Effort	0.001-0.15	0.002-0.3	2	6.25	-
		0.8145-1.1145	1.629-2.229	32	100	

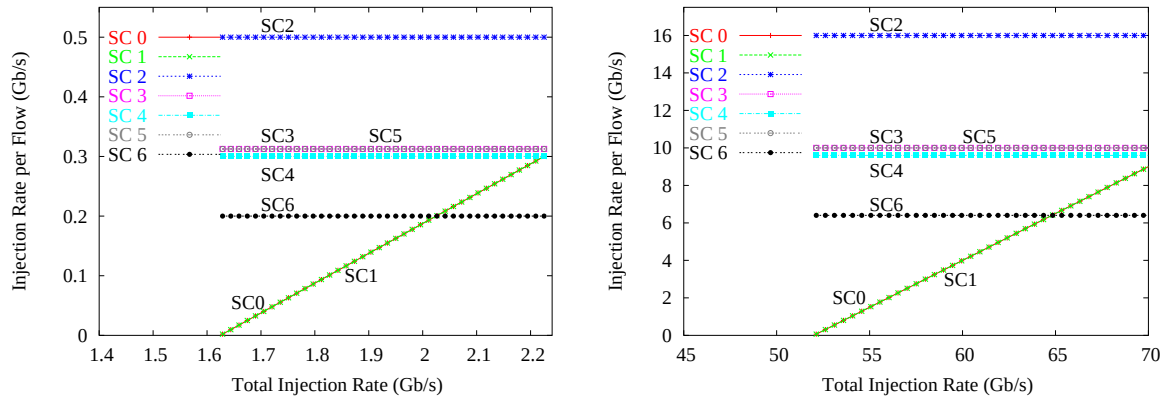


Figure 4: Injected traffic per SC for (a) the single switch network and (b) the multistage network.

5.3 Scheduler Configuration

The simulations performed compare the VC arbitration table scheduler and the MinBW scheduler. For the sake of simplicity, a table of 32 entries has been used in the simulations for the VC arbitration table scheduler approach. The table is equal in all the egress links. This situation responds to the QoS model where the table is fixed in advance and just offers a priority scheme, or it can emulate the network status after a transient period where connections have being established and the table has been modified attending to the connection requirements.

Table 1 shows the number of entries assigned to each VC, the proportion of entries that this represents, and the maximum separation of two consecutive entries in the case of the VCs assigned to DBTS traffic. In the case of the MinBW scheduler the bandwidth proportions assigned to each VC match up with the proportion of entries used in the VC arbitration table. The table configuration is shown graphically in Figure 5.

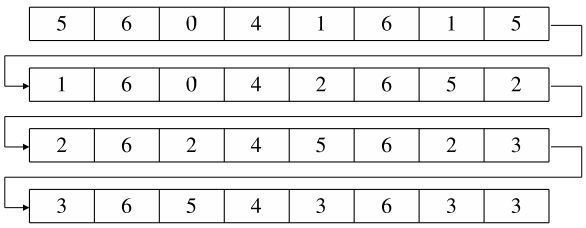


Figure 5: The VC arbitration table configuration.

5.4 Implementation of the Minimum Bandwidth Egress Link Scheduler

As a first approximation of the MinBW scheduler we have implemented the Weighted Fair Queuing algorithm (WFQ) [6]. WFQ is a work conserving algorithm that distributes the bandwidth among different flows according to a configurable set of weights taking into account the variable packet size.

The WFQ algorithm tracks the set of flows, in our case VCs, that are active in each instant. A VC is considered active if it has any packet to be transmitted. The set of active VCs is used to compute the *virtual finishing time* in which a packet would have been completely transmitted if a bit-by-bit scheme had been used. When a packet arrives at the output queues it is stamped with its virtual finishing time. The server is work conserving and serves packets in an increasing order of timestamp, taking into account only those packets that are at the head of a VC queue.

This algorithm fits the required properties of the MinBW scheduler well, except the *memoryless* one. The reason for this, is that in the WFQ algorithm the active set of VCs is determined only by whether there is a packet to transmit or not, including those that do not have enough credits to transmit the packet at the head of its queue. This means that the egress bandwidth may be allocated to VCs that are not allowed to transmit any packet due to the credit-based flow control, with those VCs taking advantage later of that bandwidth when credits are available. And this does not fit the *memoryless* property.

To solve this problem we propose a new version of the WFQ algorithm that we have called Weighted Fair Queuing Credit Aware. This algorithm is a modification of the WFQ algorithm that takes into account the available credits of each VC. The WFQ Credit Aware works on the same way than the WFQ algorithm except for these aspects:

- A VC is active only when it has some packet to be transmitted and the packet that is at the head of the queue of the VC has enough credits to be transmitted.
- A VC is inactive if it does not have any packet to be transmitted or the packet at the head of the VC queue does not have enough credits to be transmitted.
- The set of active VCs can change when a packet arrives at a VC, when a packet leaves the VC or when a credit packet arrives at the network element.
- When a VC is inactive because of lack of credits and receives enough credits to be able to transmit again, the packets in that VC are restamped as in the WFQ algorithm as if they had arrived in that instant.

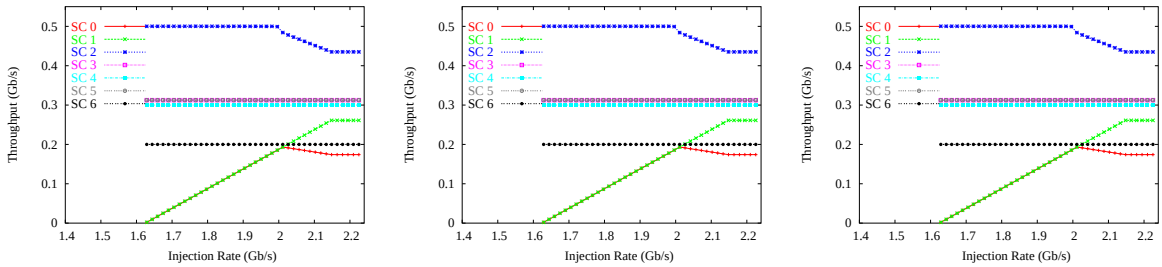
With these modifications the scheduler does not allocate bandwidth to a VC that is not allowed to transmit any packet due to the credit-based flow control. The resulting algorithm is a possible implementation of the MinBW scheduler that fits all the required properties.

5.5 Simulation Results

Figure 6 and Figure 7 show the bandwidth and average latency that the different service classes obtain in the single switch network. Figure 8 and Figure 9 show those results in the multistage network. Those figures show the network performance while the injection of best-effort traffic increases and consequently the total load of the network grows too. The total injection rate reaches a higher level than the link bandwidth. The behavior of the VC arbitration table scheduler (referred in the figures as *Table*) and the MinBW scheduler with the two possible implementations (referred in figures as *WFQ* and *WFQ Credit Aware*) are tested. We shall now study each case in turn.

5.5.1 Single Switch Network

As we want to compare the different approaches avoiding any type of possible interferences, we have used the very simple and controllable network shown in Figure 2. Figure 6 shows the bandwidth that the different service classes obtain. As can be seen the three schedulers yield almost the same results. The total injection rate and the throughput of the network are equivalent until the maximum network throughput is reached. When saturation point is reached the different service classes are treated in a different manner in function of their reserved bandwidth and the traffic that they are actually injecting.



(a) Table (b) WFQ (c) WFQ Credit Aware

Figure 6: Throughput obtained in the single switch network.

Note that all the service classes obtain at least the minimum bandwidth that they have reserved. Those service classes that inject more traffic than the bandwidth they have reserved obtain their minimum plus the result of redistributing the bandwidth left over by SC6 and SC4. The theoretical assignment of bandwidth and the real throughput obtained with the VC arbitration table of the unique switch of the network, at maximum network load are shown in Table 2. Note that SC6 injects much less than it has reserved. This unused bandwidth is redistributed among the other VCs proportionally to the minimum bandwidth reserved for each one. The SC4 obtains all the bandwidth required and more, this additional bandwidth being redistributed again between SC0-SC2.

Table 2: Theoretical assignment of bandwidth and real throughput obtained with the VC arbitration table of the unique switch of the network.

TC	Injected	Reserved	Redist. SC6	Redist. SC4	Assigned	Obtained
6	0.2	0.5	-0.3	0	0.2	0.2
5	0.3125	0.3125	0	0	0.3125	0.3125
4	0.3	0.25	0.0857	-0.0357	0.3	0.3
3	0.3125	0.3125	0	0	0.3125	0.3125
2	0.5	0.3125	0.1071	0.0179	0.4375	0.4351
1	0.3	0.1875	0.0643	0.0107	0.2625	0.261
0	0.3	0.125	0.0429	0.0071	0.175	0.174
	2.225	2	0	0	2	1.9951

Results show that the three methods considered have the same behavior. They are able to provide flows with bandwidth guarantee with the same accuracy. Note also that it is possible to have different levels of best-effort traffic (SC0 and SC1).

As was stated, we have also measured the latency obtained for each service class. These results are shown in Figure 7. Contrary to what happens with the throughput results, the latency results depend on the used scheduler. The WFQ presents higher latencies for SC3 to SC6, because these service classes are affected negatively by the service classes that are often not allowed to transmit any packet due to the credit-based flow control (those that inject more traffic than the throughput they are able to obtain). These service classes take advantage of the fact that the WFQ does not fit the *memoryless* property.

Note that SCs from 0 to 2 have a very high average latency when the network is heavily loaded. This is due to the fact that these service classes obtain a lower throughput than the injected traffic. Therefore, the packets are stored in the network elements for a long time and the average latency grows.

The results also show that when the VC arbitration table scheduler is used, SC3 obtains a higher average latency than SC5. Both SCs have the same number of entries in the table, but distributing appropriately those entries it is possible to get better and more delimited latencies. In the MinBW scheduler both service classes obtain similar results, because with this scheduler only bandwidth reservations can be made, as it is unable to differentiate traffic, taking into account any latency requirement. Note that when the saturation point is reached the DBTS traffic (SC4-SC6) obtain lower latencies using the VC table arbitration scheduler than with the MinBW scheduler.

Therefore, results show that in order to be able to provide latency guarantee an injection control (e.g. token bucket) must be used in order to avoid any flow exceeding the bandwidth it has been assigned. Note that in some cases of Figure 7 the method WFQ Credit Aware obtains lower average latency than the VC arbitration table. However, this is not a bad result for the table approach because the flows are obtaining what they have guaranteed, which matches up with what they requested. In this way, the

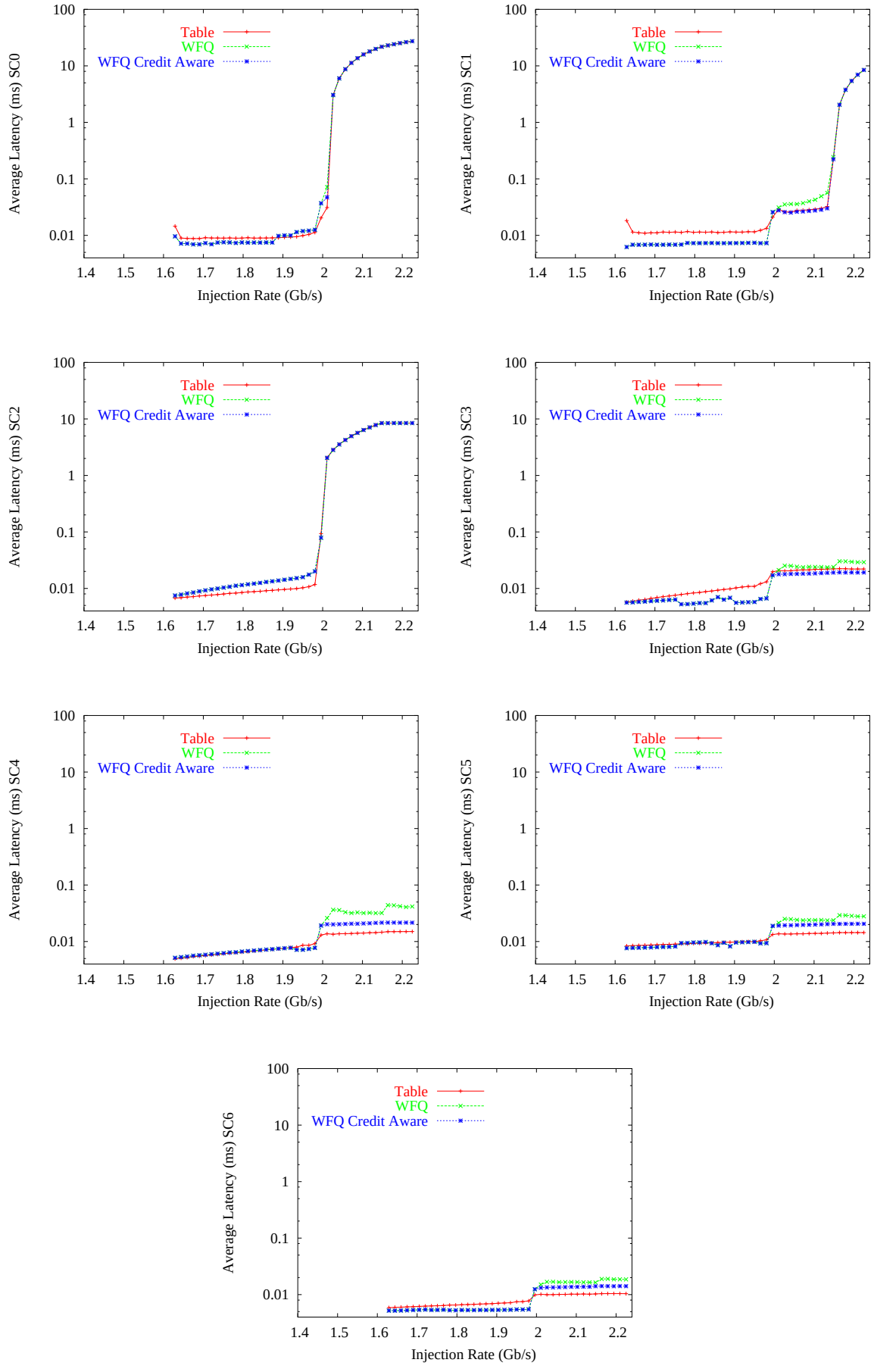
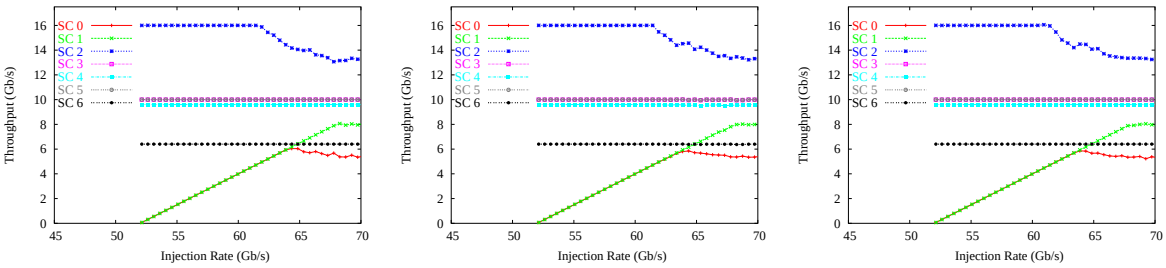


Figure 7: Average latencies obtained per SC in the single switch network.

WFQ Credit Aware approach obtains very good latency results but without providing guarantees.

5.5.2 Multistage Network

In order to have results in a more realistic network, the same experiments have been run in the multistage network. Figure 8 shows the bandwidth that the different service classes obtain.



(a) Table (b) WFQ (c) WFQ Credit Aware

Figure 8: Throughput obtained in the multistage network.

Although similar results are obtained with this network, some differences must be noticed. Saturation point is attained before the maximum link bandwidth is reached. This is because there is some network congestion. In spite of this fact, those service classes that inject more traffic than they have reserved obtain at least their minimum bandwidth reservation. When the network load is low or medium these flows get the bandwidth they are injecting. However, when the best-effort traffic increases, the network load reaches a high level and no all flows obtain what they are injecting any more. As was stated for the single switch network, the remaining bandwidth is distributed among the flows that are sending more traffic than they previously requested. For instance, SC2 has only reserved 10 Gbps and obtains more than 12 Gbps. Note again that with the three methods the same throughput results are obtained. All of them are equally able to provide bandwidth guarantee.

We have also measured the average latency for each service class. These results are shown in Figure 9. In this case, the difference between using the WFQ implementation of the MinBW scheduler and using the WFQ Credit Aware is more noticeable. The WFQ presents very high latencies for SC3 to SC6. As is already known, the WFQ is not appropriate as a MinBW implementation. Again, these results show that if a service class injects more traffic than it has reserved the network may be not able to provide the latency requirements that the service class requested.

Note that in the single switch network the traffic is addressed to the same destination. Due to this fact, in that case all the incoming traffic requests the same output link. However, in the multistage network the traffic is uniformly distributed among all the possible destinations. Therefore, the load and congestion level of the output links

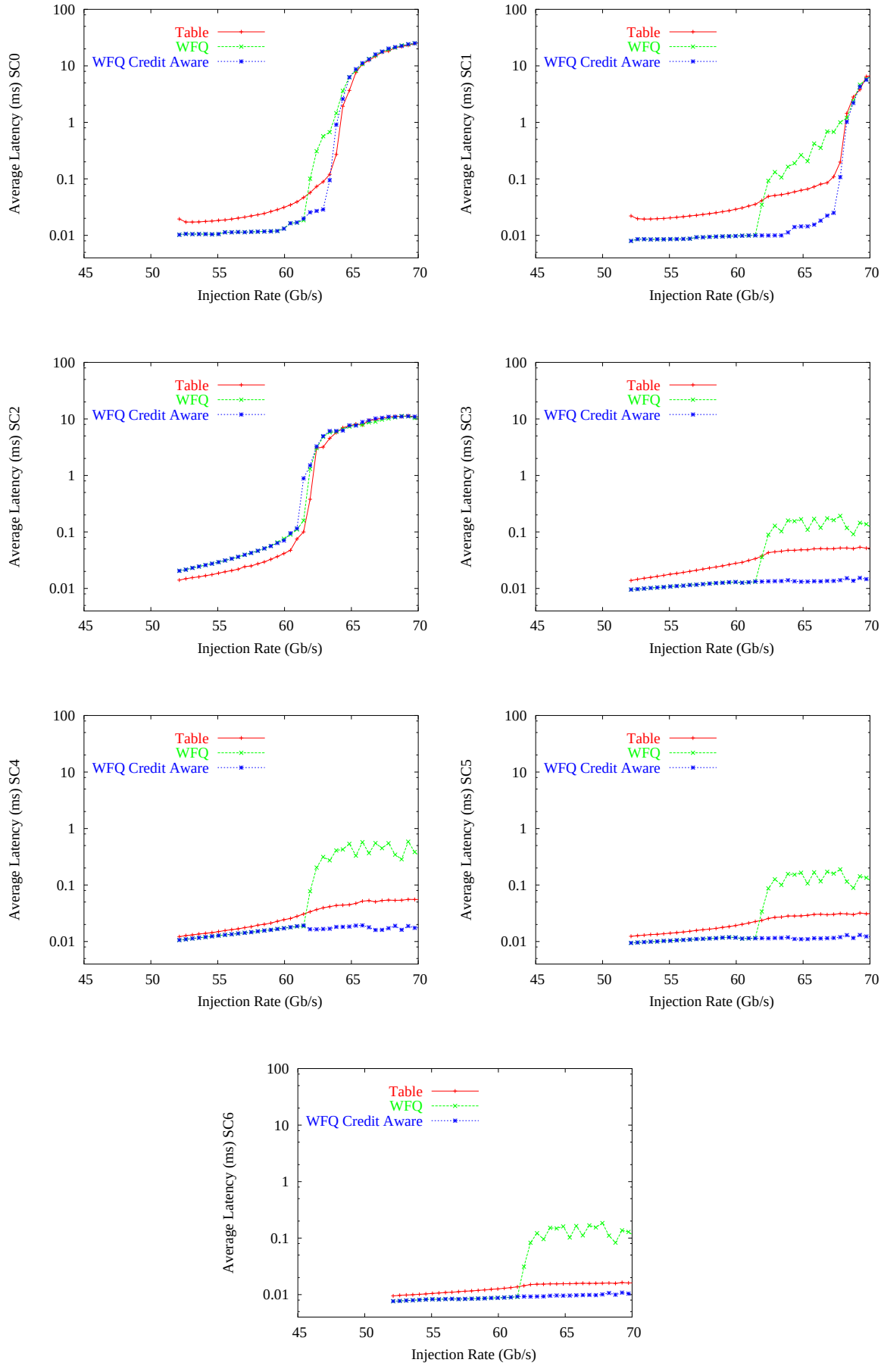


Figure 9: Average latencies obtained per SC in the multistage network.

are lower in the multistage network. This effect affects the average latency results. For instances, it can be clearly seen on the figures for SC5 when the WFQ Credit Aware proposal is used: When the network is heavily loaded, the average latency is lower in the multistage network than in the single switch network, although the average number of hops is greater for the multistage network.

In this case the average latency obtained for DBTS traffic using the VC table arbitration scheduler is higher than that obtained with the MinBW scheduler (using the WFQ implementation). This does not mean that it is a worse result, since they get a latency according to the entries in the VC arbitration table, which is what they requested. This approach provides flows with latency guarantee according to their requirements. If they do not need to obtain a lower latency other traffic classes can take advantage obtaining a lower latency than in the MinBW scheduler case.

6 Conclusions

In this paper, we have proposed several methods to use the AS mechanisms to provide applications with QoS. Specifically, we have shown how to provide QoS based on bandwidth and latency requirements. We have proposed to segregate the traffic into different service classes. These service classes must be processed appropriately by the egress link scheduler to obtain their QoS requirements.

We have shown that if we want to provide any type of guarantee a connection admission control must be used to manage correctly the available resources. Moreover, if we want to provide applications with bandwidth and latency guarantee we have proposed to use the AS VC arbitration table scheduler, and to assign dynamically the arbitration table entries to the VCs according to the requirements that the traffic flows using those VCs have. However, if we only want to provide applications with bandwidth guarantee (no latency guarantee) or to provide a differentiated treatment to the different service classes, we can also use the MinBW scheduler.

We have tested the behavior of the proposed methodology by simulation using CBR traffic with all the packets of the same size. For comparison, we have proposed an implementation of the MinBW scheduler that fills all the required properties, and have called this algorithm WFQ Credit Aware.

The results obtained show that the injection flows must be controlled in order not to transmit more bandwidth than they have reserved. AS provides the token bucket method to perform that. If the flows are not controlled the bandwidth requirements can be guaranteed but not the latency requirements.

Results also show that the VC arbitration table is the only specified mechanism that can be used to provide both bandwidth and latency guarantee. However, the WFQ Credit Aware method obtains very good results for both bandwidth and latency, but it can not provide any kind of guarantee. Therefore, this algorithm is very good for the implementation of a priority scheme in AS with differentiated services, that are not intended to provide guarantee.

As future work, we are focusing our attention upon several aspects to improve AS arbitration table behavior. We are studying different ways to solve the problem that arises when variable packet size is considered. Moreover, we would like to test our proposals on an AS commercial product as soon as we have one available, in order to verify the practicality of our model.

Acknowledgments

The authors would like to thank Frank Olaf, Sven-Arne Reinemo, and Olav Lysne from the Simula Resarch Laboratory of Norway, their useful comments which have helped to considerably improve the quality of this paper.

Bibliography

[1] Advanced Switching Interconnect Special Interest Group. *Advanced Switching Core Architecture Specification. Revision 1.0*, December 2003.

[2] F.J. Alfaro, J.L. Sánchez, and J. Duato. A New Proposal to Fill in the InfiniBand Arbitration Tables. In *Proceedings of IEEE International Conference on Parallel Computing (ICPP'03)*, pages 133 – 140, October 2003.

[3] F.J. Alfaro, J.L. Sánchez, and J. Duato. QoS in InfiniBand Subnetworks. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 15(9):194–205, September 2004.

[4] S. Blake, D. Back, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. Internet Request for Comment RFC 2475, Internet Engineering Task Force, December 1998.

[5] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. Internet Request for Comment RFC 1633, Internet Engineering Task Force, June 1994.

[6] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *Journal of Internetworking Research and Experience*, 1, 1990.

[7] Advanced Switching Interconnect Special Interest Group. `http://www.asisig.org`.

[8] PCI Special Interest Group. `http://www.pcisig.org`.

[9] PCI Special Interest Group. *PCI Express Base Architecture Specification. Revision 1.0a*, April 2003.

[10] J. Pelissier. Providing Quality of Service over Infiniband Architecture Fabrics. In *Proceedings of the 8th Symposium on Hot Interconnects*, August 2000.

[11] S.A. Reinemo, F.O. Sem-Jacobsen, T. Skeie, and O. Lysne. Admission control for diffserv based quality of service in cut-through networks. In *In Proceedings of the 10th International Conference on High Performance Computing (HiPC 2003). Hyderabad, India*, pages 118–128, December 2003.