

ABR ARCHITECTURE AND SIMULATION FOR AN INPUT-BUFFERED AND PER-VC QUEUED ATM SWITCH

Matthias Bossardt

Sueng-Yong Park

John W. Lockwood

Sung-Mo Kang

ipoint@ipoint.vlsi.uiuc.edu

Department of Electrical and Computer Engineering

University of Illinois at Urbana-Champaign

405 N. Mathews Avenue

Urbana, IL 61801, U.S.A.

ABSTRACT

This paper proposes an innovative concept, called *virtual output queue*, to support available bit rate (ABR) traffic on an input-buffered, per virtual circuit (VC) queued switch. This technique allows ABR models developed for output-buffered systems to be migrated to an input-buffered system.

In order to evaluate the virtual output queue and to compare different ABR algorithms, a simulator of the ATM testbed at the University of Illinois has been enhanced with ABR functions. This paper provides simulation results for the input-buffered variation of the ERICA+ algorithm.

INTRODUCTION

ABR is designed for computer data transmission in ATM networks. For an ABR connection, the source receives feedback from the network and adjusts its rate according to this information. The feedback information is carried within special cells, called Resource Management cells (RM cells). RM cells are periodically inserted into the stream of ABR data cells. As a consequence of the feedback mechanism, cell loss is minimized and the available bandwidth is shared fairly by the ABR connections.

The first section of this paper presents the *virtual output queue*, a technique which approximates the length of an output-buffered queue for ABR traffic on an input-buffered switch. The next section deals with the general implementation of ABR fair-share rate algorithms with the virtual output queue. The last section shows simulation results for ERICA+ with current cell rate (CCR) measurement. More simulations have been presented in [1] and [2], where the DMRCA algorithm [3] and ERICA+ [4] have been adapted to an input-buffered switch architecture.

Input-Queuing

The iPOINT switch [5], [6] developed at the University of Illinois features an input-buffered architecture. The main advantage of this design is its high scalability. For an input-buffered switch, the memory bandwidth is independent of the number of ports. At most, only one cell may be written into the memory and only one cell may be read from the memory per cell slot [7]. Through the use of a cell scheduler, the switch fabric needs only to provide crossbar functionality [8].

The major shortcoming of an input-queued switch architecture is the head-of-line (HOL) blocking problem, that occurs when simple FIFO queues are used. For the iPOINT switch, the iiQueue has been developed, featuring a three-dimensional queue (3DQ). The 3DQ sorts the cells according to three criteria: their connection, their priority, and their output port [9]. This provides near-100% link utilization and supports QoS for multiple traffic types.

THE VIRTUAL ABR OUTPUT QUEUE CONCEPT

This paper presents a concept that allows ABR algorithms, designed for an *output*-buffered switch to be used in an *input*-buffered and *per-VC* queued switch. This technique uses in-band communication to transmit data among the different modules of the switch. The supplementary information is carried in unused bytes of the RM cell and therefore does *not* affect the throughput of the switch. The computational complexity of the virtual output queue is independent of the number of VCs.

The Problem of Input Queuing

ABR algorithms need to measure the congestion of an outgoing link. Two schemes are possible: measurement of the link utilization and/or measurement of the ABR queue length. The most powerful algorithms use queue length measurement.

It is not simple to characterize the outgoing ABR queue length on an input-buffered switch with per-VC queuing. Consider the simple network in Figure 1, featuring one switch, three sources, and one destination. Traffic is flowing from the three ABR sources (S1, S2, and S3) to the corresponding destination (D). Three ports of the switch are connected to the three sources (S1, S2, and S3). The fourth port of the switch is connected to the destination (D).

When the sources are not internally bottlenecked, each is able to transmit at the maximum speed of the link. In this case, the link connected to the destination becomes congested. For an input-buffered switch, this bottlenecked link triggers queue growth at each input port with incoming traffic.

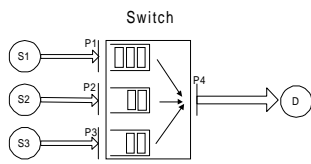


Figure 1: Input-buffered switch.

The ABR algorithms studied in [3] and [4] are designed to be located where congestion takes place. In the example, the switch is congested at port 4.

In an output-queued switch, the queue of port 4 would grow and could be measured easily by the ABR algorithm, implemented at the same port. In an input-queued switch, however, measurement of the queue length is distributed over several input ports, as shown in the example. In other words, the buffer for the cells heading to a certain link is partitioned into several queues located in physically different modules.

Virtualizing the Queue

The goal of this work is to apply existing ABR algorithms to an input-buffered switch architecture. Since these algorithms assume a single FIFO output queue for ABR traffic, we have to take some steps in order to simulate a similar architecture to the ABR algorithm on our switch. Figure 2 illustrates the problem of mapping the switch designed for (a) to a system like (b) for which the ABR algorithms have been developed.

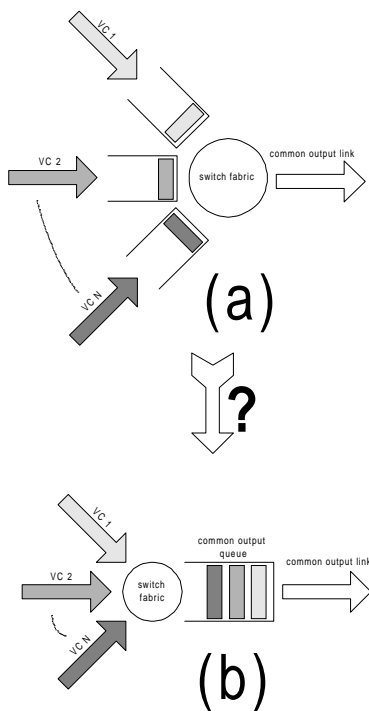


Figure 2: The problem of distributed queues.

Transmitting Queue Length Information within the Switch

The idea of the virtual output queue is to collect the queue length information of all ABR VCs having a common output port and to transmit this information from the ingress module of the source input ports to the egress module of the common output port. Figure 3 illustrates this flow of information. We assume a configuration as in Figure 1, where the sources are connected to ports 1, 2, and 3. The common destination is at port 4.

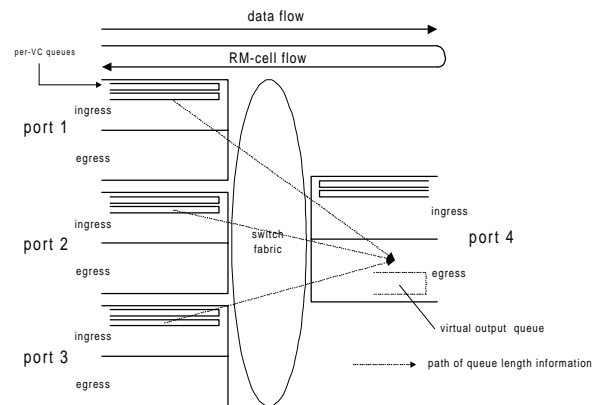


Figure 3: The virtual output queue.

A key observation is that the RM cell is a good carrier for this information:

- Some reserved bytes of the RM cells are constant. They do not carry any information. Therefore, it is possible to replace bytes 22 and 23 of the RM cell with queue length information when the RM cell is at the ingress module of the switch. The information is used uniquely within the switch. In order to avoid violating the ATM Forum specifications [10], the value $6A6A_{(\text{hex})}$ is written again in bytes 22 and 23 at the egress module before the RM cell exits the switch.
- No additional overhead is produced since no additional cells are generated within the switch.
- No additional wiring between the ports is needed, since the RM cell is switched in the same way as a user data cell of the same VC.

Note that the queue length (QL) field of the RM cell is not related to the virtual queue length. It represents the maximum number of cells currently queued for this connection among those network elements supporting this parameter [11].

Assembling the Virtual Output Queue

The virtual output queue pretends that there is a single FIFO queue for all ABR traffic at the egress module of a port, although the cells are actually stored at the ingress module of one or more ports. The desired information at the egress is the length of the virtual output queue.

$$\text{Virtual_Output_QL} = \sum_{i \in \text{ABR_VC}} \text{QL}(\text{VC}_i) \quad (1)$$

where QL is a queue length.

A naive way to transmit the queue length would be to write the per-VC queue length directly into the RM cell. At the egress, the queue length for each of the ABR VCs would be stored separately in a table. Scalability in terms of number of VCs and speed would be very limited. A significant amount of memory would be needed to store the per-VC queue lengths at the egress modules.

This paper proposes a solution that uses differential information. Instead of writing the absolute length of the per-VC queue into the RM cell, the difference of the current queue length ($\text{QL}_{\text{new}}(\text{VC}_i)$) and the queue length at the time when the last RM cell of VC_i arrived at the ingress ($\text{QL}_{\text{old}}(\text{VC}_i)$) is stored within the RM cell.

$$\Delta\text{QL}(\text{VC}_i) = \text{QL}_{\text{new}}(\text{VC}_i) - \text{QL}_{\text{old}}(\text{VC}_i) \quad (2)$$

$\text{QL}_{\text{new}}(\text{VC}_i)$ is then written into $\text{QL}_{\text{old}}(\text{VC}_i)$.

In order to calculate the length of the virtual output queue, the egress side totals the per-VC queue lengths. Each time a value $\Delta\text{QL}(\text{VC}_i)$ arrives in a forward RM cell at the egress, the Virtual_Output_QL is updated as shown below.

$$\text{Virtual_Output_QL}_{\text{new}} = \Delta\text{QL}(\text{VC}_i) + \text{Virtual_Output_QL}_{\text{old}} \quad (3)$$

From Equation (3), we can see that only one addition must be performed when a forward RM cell arrives at the egress side. As we also notice from this equation, the calculation complexity of the virtual output queue length is independent of the number of ABR VCs.

Note that the Virtual_Output_QL is delayed, since $\Delta\text{QL}(\text{VC}_i)$ is written into the RM cell when the RM cell arrives at the per-VC queue of the ingress. The Virtual_Output_QL is updated when this RM cell arrives at the egress.

$\Delta\text{QL}(\text{VC}_i)$ is bounded by the interval $[-\text{MaxQueueLength}(\text{VC}_i), \text{MaxQueueLength}(\text{VC}_i)]$. Thus, depending on the value of $\text{MaxQueueLength}(\text{VC}_i)$, more than two bytes of the RM cell might be needed to carry the $\Delta\text{QL}(\text{VC}_i)$ information.

Differential information is sensitive to errors. If $\Delta\text{QL}(\text{VC}_i)$ is corrupt, the error will propagate and affect all the future calculations of the virtual output queue length. This problem does not occur for systems with switch fabrics that account for all cell loss (such as iPOINT). While RM cells can be lost at arrival at the

switch because of queue overflow, the queue will know that they were dropped. $\Delta\text{QL}(\text{VC}_i)$ is *not* written into the RM cell and $\text{QL}_{\text{old}}(\text{VC}_i)$ is *not* updated before it is assured that the queue has enough space to accommodate the incoming cell.

ABR Functional Blocks

The following paragraphs present how the ABR algorithms must be partitioned into hardware logic and where the different parts would be implemented.

Let us first consider a simplified input-buffered and per-VC queued switch, which is represented in Figure 4. This example has one ABR connection entering the switch at port 1 and leaving it at port 2. This is sufficient to explain the principle and can be easily generalized for an arbitrary number of connections. Moreover, only two ports of the switch are represented in the figure. Each port includes two blocks (1 and 2) containing parts of the ABR algorithm.

From the standpoint of a connection, block 1 is split into subblocks 1a and 1b. Each modifies the RM cell at the ingress module of a port. Depending on the direction of the RM cell, either 1a or 1b is executed. The DIR field of the RM cell indicates whether the cell travels from the source to the destination (forward RM cell: $\text{RM.DIR}=0$) or vice versa (backward RM cell: $\text{RM.DIR}=1$) (Figure 4).

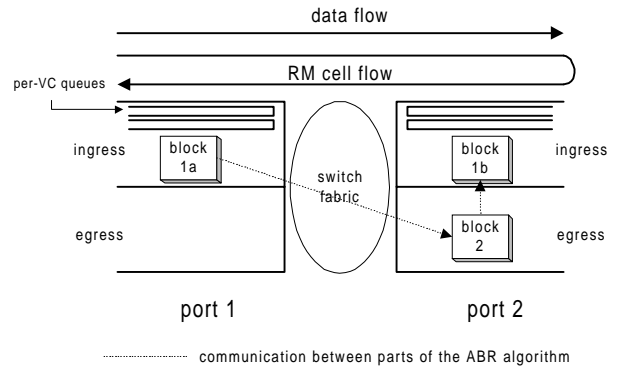


Figure 4: Functional blocks of the ABR algorithms.

Subblock 1a generates differential queue length information and writes it into the forward RM cell.

Subblock 1b computes the feedback according to a fair-rate allocation algorithm (e.g., ERICA+) and writes it into the backward RM cell (Figure 5).

Block 2 collects the differential queue length information and calculates the virtual output queue length. It further measures and calculates values to determine the fair-share rate as proposed by a fair rate allocation algorithm. Block 2 might also extract other values from forward RM cells (Figure 6).

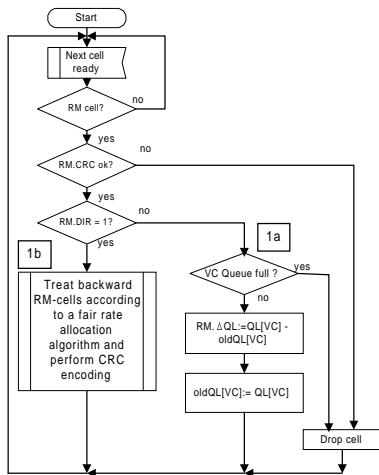


Figure 5: Block 1.

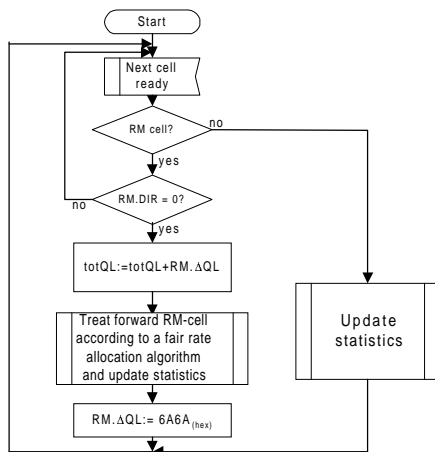


Figure 6: Block 2.

Influence of the Virtual Output Queue

The length of the virtual output queue is an estimation of the number of ABR cells stored in the switch for a given link. Since the differential queue length is written into the RM cell before it enters the per-VC queue, the information transmitted by the RM cell is delayed as a function of the per-VC queue lengths. Moreover, the per-VC queue length is only sampled when an RM cell arrives at the switch. The goal of this section is to determine the penalty of these inaccuracies.

In order to determine the magnitude of this performance penalty, simulations with virtual output queue are compared to simulations where the queue length is determined without delay.

SIMULATION RESULTS

A simulator of the iPOINT switch [12] has been enhanced with ABR functionality. For the simulations in this paper, a standard fair-rate allocation algorithm has been implemented. The algorithm is called the enhanced Explicit Rate Indication for Congestion Avoidance (ERICA+) algorithm [4], [13]-[15]¹. Its simulation [1], [2] has shown that the performance of ERICA+ is much better when the current cell rate (CCR) is measured at the switch than when the CCR field of the RM cells is used. This paper shows the simulation results of ERICA+ with CCR measurement. The number of ABR connections at a certain link is assumed to be known, since this value can be communicated to the corresponding modules by control cells. To evaluate the performance of the algorithm together with the virtual output queue, this paper simulates the standard configuration given in Figure 7. The simulation uses ABR sources and destinations that are never internally bottlenecked.

Parking Lot Configuration

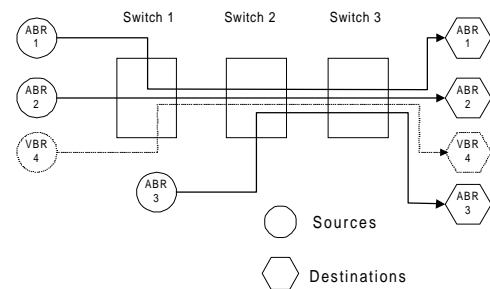


Figure 7: Parking lot configuration.

In the parking lot configuration, one connection (VC 3) passes across fewer switches than the others do. The bottleneck is at the output of switch 2. If the ABR algorithm is fair, each ABR connection gets one third of the available ABR bandwidth. Algorithms with fairness problems will allocate a higher rate to the connection that travels across fewer switches (VC 3). VC 4 transmits VBR traffic. Its effect is to modulate the available ABR bandwidth. Figure 8 shows the applied VBR waveform. Table 1 contains the values of the parameters.

The link length between switches is set to 1 km for a campus area network (CAN) and to 1000 km for a wide area network (WAN).

¹ Two slightly different versions of ERICA+ can be found in [4] and [13]. For the simulations, the version of [4] was applied. In [13], ER is calculated as $ER := \text{Max}(\text{MaxAllocPrevious}, \text{VC_Share})$, whereas in [4], ER is calculated as $ER := \text{Max}(\text{MaxAllocPrevious}, \text{Fair_Share}, \text{VC_Share})$.

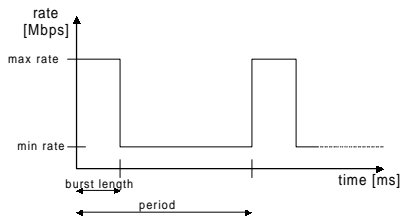
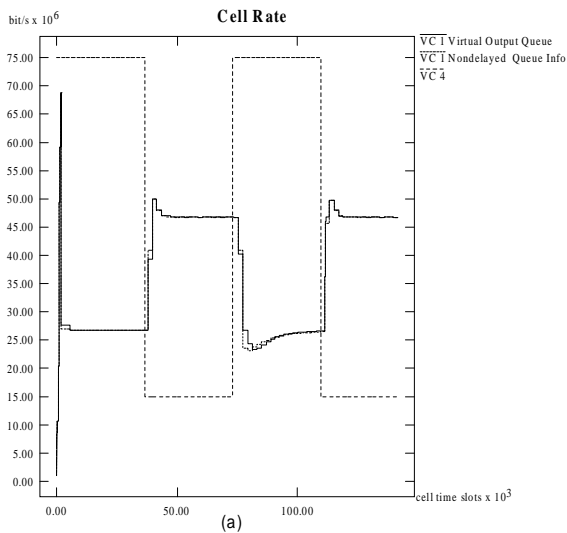


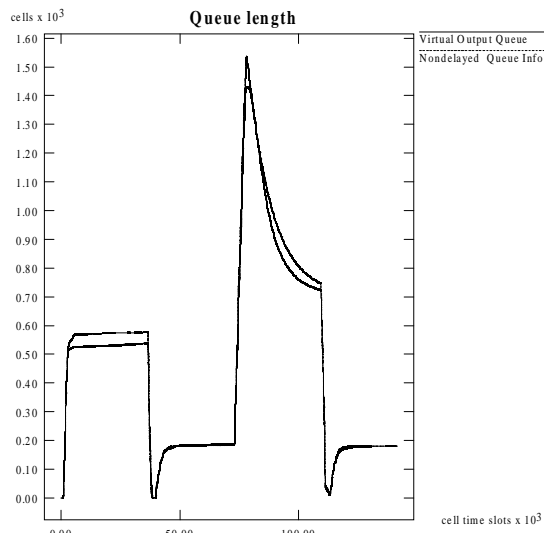
Figure 8: VBR waveform.

Table 1: VBR parameters.

Period [ms]	Burst length [ms]	Max-to-min bit rate ratio	Min. rate [Mbps]
200	100	2 - 8 (default:5)	15



(a)



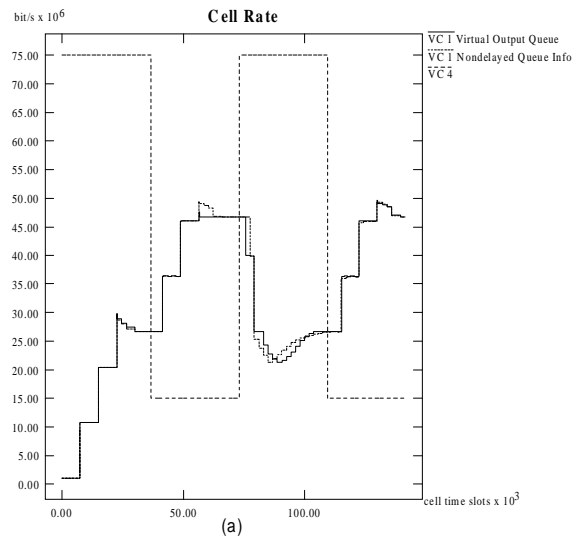
(b)

Figure 9: ERICA+ : influence of virtual output queue (queue length: switch 2) in CAN.

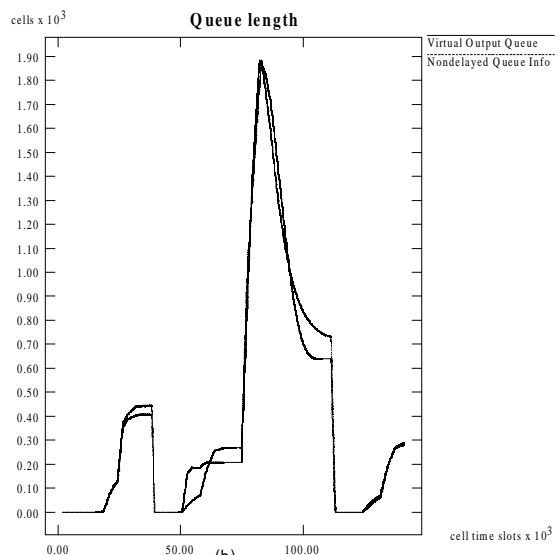
Figure 9 (a) compares the cell rate of VC 1 when the virtual output queue is applied to the case where

nondelayed queue length information is available. The time axis is measured in “cell time slots.” At OC-3 speed (155 Mbps) one “cell time slot” equals 2.7 μ s. In addition to the VC 1 cell rates, we also show the rate of the VBR traffic (VC 4). A campus area network (CAN) is simulated, where the link length between switches is 1 km. We observe that the influence of the virtual output queue is negligible.

Figure 9 (b) shows the number of stored ABR cells (queue length) in switch 2 when the virtual output queue is applied as well as the case where non-delayed queue length information is available. The difference in the number of stored cells is also very small.



(a)



(b)

Figure 10: ERICA+ : influence of virtual output queue (queue length: switch 2) in WAN.

Figure 10 shows the same simulations as Figure 9, except that a wide area network (WAN) is simulated. In this case the link length between switches equals 1000

km. The long links result in a larger feedback delay, and we observe that the convergence to the fair-share rate is slower than in the CAN (Figure 9). The VC 1 cell rate grows from an initial value of 1 Mbps, when the first RM cell arrives back at the source. The cells travel 1 km within 5 μ s. It takes the first RM cell about 20 ms in this configuration until to return to the source, since initially there are no delays by queues.

We notice also that due to the larger feedback delay, the queues grow larger in the WAN. However, as with the CAN, the number of stored ABR cells (queue length) in switch 2 is nearly identical when the virtual output queue is applied as when the queue length information is non-delayed.

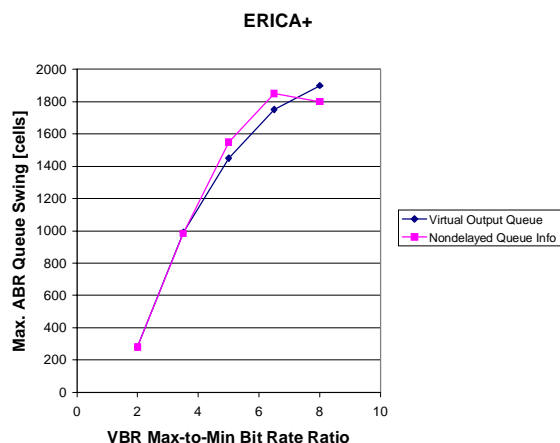


Figure 11: ERICA+ : maximum ABR queue length swing as a function of the VBR Max-to-Min Bit Rate Ratio in a CAN.

Figure 11 shows the maximum ABR queue length swing as a function of the VBR max-to-min bit rate ratio. We notice that the maximal ABR queue swing grows with the increase of the max-to-min bit rate ratio. The simulation with virtual output queue shows no significant difference to the case with nondelayed queue information.

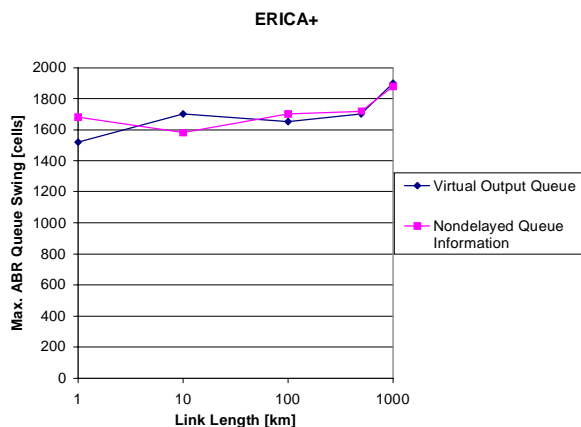


Figure 12: ERICA+: maximum ABR queue length swing as a function of the link length between the switches.

Using the default modulation of the VBR source (Table 1), Figure 12 shows the maximal queue swing as a function of the link length. We notice that the virtual output queue has no negative influence and can be used in large networks.

CONCLUSION

In this paper, we have presented and evaluated an innovative concept to provide ABR service in an input-buffered and per-VC queued switch. The virtual output queue concept enables us to apply existing fair-rate allocation algorithms to input-buffered, per-VC queued ATM switches without generating additional communication overhead. The computation complexity of the implementation is independent of the number of ABR connections and is sufficiently compact to be implemented in only a few thousand gates.

Simulation results show that the virtual output queue works well with a standard fair-rate allocation algorithm such as ERICA+.

REFERENCES

- [1] Matthias Bossardt, "Available bit rate architecture and simulation for an input-buffered and per-VC queued ATM switch," Diploma thesis, University of Illinois at Urbana-Champaign, Urbana, U.S.A., and Swiss Federal Institute of Technology, Lausanne, Switzerland, February 1998.
- [2] Matthias Bossardt, "Simulations of ERICA+ with per virtual connection current cell rate measurement," University of Illinois at Urbana-Champaign, Urbana, U.S.A., <http://ipoint.vlsi.uiuc.edu/abr/>.
- [3] F. M. Chiusi, Y. Xia, and Vijay P. Kumar, "Dynamic max rate control algorithm for available bit rate service in ATM networks," in *Proceedings of IEEE GLOBECOM*, 1996, pp. 2108-2117.
- [4] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and R. Viswanathan, "The ERICA switch algorithm for ABR traffic management in ATM networks, part I: description," <http://www.cis.ohio-state.edu/~jain/>.
- [5] J. W. Lockwood, S. M. Kang, S. G. Bishop, H. Duan, and A. Hossain, "Development of the iPOINT testbed for optoelectronic asynchronous transfer mode networking," in *International Conference on Information Infrastructure*, April 25-28, 1996, Beijing, China, pp. 509-513.
- [6] J. W. Lockwood, H. Duan, J. J. Morikuni, S. M. Kang, S. Akkineni, R. H. Campbell, "Scalable optoelectronic ATM networks: the iPOINT fully functional testbed," *IEEE Journal of Lightwave Technology*, pp. 1093-1103, June 1995.
- [7] H. Duan, J. W. Lockwood, and S. M. Kang, "An efficient input queuing and cell scheduling scheme for scalable ultra-broadband optoelectronic ATM switching," in *Photonics East 95 Conference* October, 1995, Philadelphia, Pennsylvania. SPIE Proceedings, Volume 2608, pp. 107-108.
- [8] H. Duan, J. W. Lockwood, and S. M. Kang, "Matrix unit cell scheduler (MUCS) for input-buffered switches," *IEEE Communication Letters*, Volume 2, Number 1, January, pp. 20-23, 1998.
- [9] H. Duan, J. W. Lockwood, S. M. Kang, and J.D. Will, "A high-performance OC-12/OC-48 queue design prototype for input-buffered ATM switches," in *IEEE INFOCOM '97*, Kobe, Japan, April 7-11, 1997, pp. 20-28.
- [10] The ATM Forum, *Traffic Management Specification, Version 4.0*, April 1996.
- [11] ITU-T, Recommendation I.371, *Traffic Control and Congestion Control in B-ISDN*, Perth, November 1995.
- [12] Sueng-Yong Park, "An Event-Driven Behavioral ATM Switch Simulator," M.S. Thesis, University of Illinois at Urbana-Champaign, Urbana, U.S.A., December 1997.
- [13] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore, "The ERICA switch algorithm for ABR traffic management in ATM networks," <http://www.cis.ohio-state.edu/~jain/>.
- [14] B. Vandalore, R. Jain, R. Goyal, and S. Fahmy, "Design and analysis of queue control function for switch schemes," ATM FORUM proposal 97-1087.
- [15] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and R. Viswanathan, "The ERICA switch algorithm for ABR traffic management in ATM networks, part II: requirements and performance evaluation," <http://www.cis.ohio-state.edu/~jain/>.