# Credit-Based Flow Control for ATM Networks

T. Blackwell, K. Chang, H. T. Kung, and D. Lin

*Division of Applied Sciences*
*Harvard University*
*29 Oxford Street, Cambridge, MA 02138*

**Abstract**

In credit-based flow control for ATM networks, switch buffer space is first allocated to each virtual circuit (VC) and then credit control is applied to the VC to prevent possible buffer overflow. Adaptive buffer allocation improves sharing by allowing dynamic allocation of buffer space to multiple VCs sharing the same buffer pool. This paper gives an overview of credit flow control and presents performance results from simulations.

## 1. Introduction

Flow control is essential for asynchronous transfer mode (ATM) networks [1] in providing "best-effort" services, or ABR (Available Bit Rate) services in the ATM Forum terminology. With proper flow control, computer users would be able to use an ATM network in the same way as they have been using conventional LANs. That is, they can use the network at any time without first negotiating a "traffic contract" with the network. Any user would be able to acquire as much network resources as are available at any given moment, and all users would compete equally for the available bandwidth.

This paper describes an efficient way of implementing flow-controlled ATM networks through the use of credit-based, per VC, link-by-link flow control [11, 12, 14].

The organization of the paper is as follows: First, motivations for per VC, link-by-link flow control are given. This is followed by an overview of the credit-based flow control approach. Then simulation results are presented for credit flow control with adaptive buffer allocation.

A version of the proposed credit-based flow control scheme has been implemented on an experimental ATM switch with 622-Mbps ports, currently under joint development by BNR and Harvard. See a companion paper in these proceedings [4].

## 2. Why Per VC Link-by-Link Flow Control?

The Flow-Controlled Virtual Connections (FCVC) approach [12], using per VC, link-by-link flow control, is different from other proposals on congestion control (see, e.g., [3, 9, 15]). Our interest in FCVC is primarily due to its effectiveness in maximizing network utilization, controlling congestion, and implementing "best-effort" or ABR services.

### 2.1. Maximizing Network Utilization

FCVC provides effective means of using fill-in traffic to maximize network utilization, as depicted in Figure 1. Using FCVC, best-effort traffic can effectively fill in bandwidth slack left by scheduled traffic with guaranteed bandwidth and latency such as video and audio. In the fill-in process, various scheduling policies can be employed. For example, high-priority best-effort traffic can be used in the fill in before the low-priority traffic.


Figure 1: Fill in bandwidth slacks with "best-effort" traffic

For effective traffic fill in, fast congestion feedback for individual VCs is needed. Measurements have shown that data [7, 13] and video [8] traffic often exhibit large bandwidth variations even over time intervals as small as 10 milliseconds. With the emergence of very high-bandwidth traffic sources such as high-speed host computers with 800-Mbps HIPPI network [2] interfaces, networks will experience further increases in load fluctuations [10]. To utilize slack bandwidth in the presence of highly bursty traffic, fast congestion feedback is necessary.

To illustrate the need of fast feedback or flow control for effective fill in, consider a simple case of maximizing the utilization of a link. As depicted in Figure 2, there are multiple VCs from the sender to the receiver sharing the link. The VC scheduler at the sender selects (when possible), for each cell cycle, a VC from which a cell will be transmitted over the link. It is intuitively clear how the scheduler should work; that is, after satisfying VCs of guaranteed performance, the scheduler will select other VCs ("fill-in" VCs), with high priority ones first, to fill in the available bandwidth of the link.

However, two additional conditions (both requiring fast flow control) must be satisfied in order to achieve effective fill in:

- First, data to be used for fill in must be "drawn" to the sender in time. That is, these fill-in VCs should try to hold in their buffers at the sender a number of cells that are ready to be forwarded. There should be sufficiently many of these cells so that they

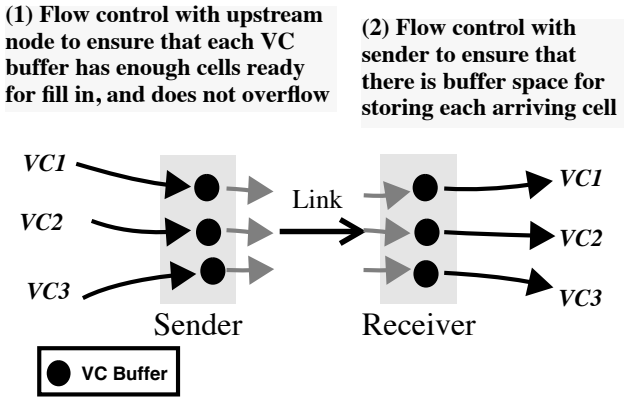


Figure 2: Two reasons for flow control, (1) at the sender and (2) at the receiver, in achieving effective traffic fill in

can fill in slack bandwidth at a high rate as soon as the bandwidth becomes available. Note that how long these cells will stay at the sender depends on the load of other VCs. When the cells of a VC are not moving out, the upstream node of the VC needs to be flow controlled to avoid buffer overflow. On the other hand, when these cells start moving out, the flow control mechanism should be able to draw in additional cells from the upstream node to fill VC buffers at the sender.

- Second, only "deliverable" traffic should be transmitted over the link in the sense that transmitted data should not be dropped at the receiver due to lack of buffer space. That is, the receiver should have buffer space for storing each arriving cell. Flow control is thus needed for the receiver to inform the sender about buffer space availability. The cost of retransmitting dropped packets increases with both the bandwidth and size of the network, such that on nationwide gigabit networks the penalty is very high.

By using link-by-link flow control, FCVC implements the required feedback at the fastest possible speed. Performance simulation [5, 11] has confirmed the effectiveness of FCVC in filling in traffic, and thus in maximizing network utilization.

## 2.2. Controlling Congestion

Another reason for FCVC is congestion control. For high-speed networks, in addition to highly bursty traffic mentioned above, there is the problem of increased mismatches in bandwidth [10]. When the peak speed of links increases in a network, so may bandwidth mismatches in the network. For example, when a 1-Gbps link is added to a network which includes a 10-Mbps Ethernet, there will be two orders of magnitude difference in their speeds. When data flows from the high-speed link to the low-speed one, congestion will build up quickly. This represents additional congestion scenarios beyond the usual congestion caused by the merging of multiple traffic streams.

The highly bursty traffic and increased bandwidth mismatches expected will increase the frequency of transient congestion. It is therefore important to ensure that transient congestion does not persist and evolve into permanent network collapse.

Using FCVC, a VC can be guaranteed not to lose cells due to congestion. When experiencing congestion, backpressure will build up quickly along congested VCs spanning one or more hops. When encountering backpressure, the traffic source of a congested VC can be throttled. Thus excessive traffic can be blocked at the boundary of the network, instead of being allowed to enter the network and cause congestion problems to other traffic.

By using "per VC" flow control, FCVC allows multiple VCs over the same physical link to operate at different speeds, depending on their individual congestion status. In particular, congested VCs cannot block other VCs which are not congested.

The throttling feature on individual VCs, enabled by FCVC, is especially useful for implementing high-performance, reliable multicast VCs. At any multicasting point involving more than a few ports, the delay before a cell is forwarded out all the ports can fluctuate greatly. It is therefore essential for reliable multicast VCs to throttle in order to accommodate the inherent high variations in their transmission speeds. Thus, the credit value can be based on the slowest port (the one with the largest queue) to ensure that no buffer will be overrun. Of course, in practice a "relatively" reliable multicast which allows some sort of time-out on blocked multicasting ports will be implemented so that a blocked port will not hold up the whole multicast VC for an unbounded amount of time.

### 2.3. Implementing "Best-Effort" or ABR Services

Flow control will enable services for hosts with high-speed network access links operating, for example, at 155 Mbps. For instance, these hosts can be offered a new kind of data communications service, which may be called a "greedy" service, where the network will accept as much traffic as it has available bandwidth at any instant from VCs under this service. FCVC can throttle these VCs on a per VC basis when the network load becomes too high, and also speed them up when the load clears. This is exactly the traditional "best-effort" service typical for hosts in LAN environments. There will be no requirements for predefined service contract parameters, which are difficult to set.

## 3. Credit-Based Flow Control

Flow control based on credits is an efficient way of implementing per VC link-by-link flow control. A credit-based flow control method generally works over each flow-controlled VC link as follows (see Figure 3). Before forwarding any data cell over the link, the sender needs to receive credits for the VC via credit cells sent by the receiver. At various times, the receiver sends credit cells to the sender indicating availability of buffer space for receiving data cells of the VC. After having received credits, the sender is eligible to forward some number of data cells of the VC to the receiver according to the received credit information. Each time the sender forwards a data cell of a VC, it decrements its current *credit balance* for the VC by one.

As a consequence of this scheme, the number of cells sent in one round trip time of the link cannot be more than the buffer allocation. Thus, small buffer allocations will limit the

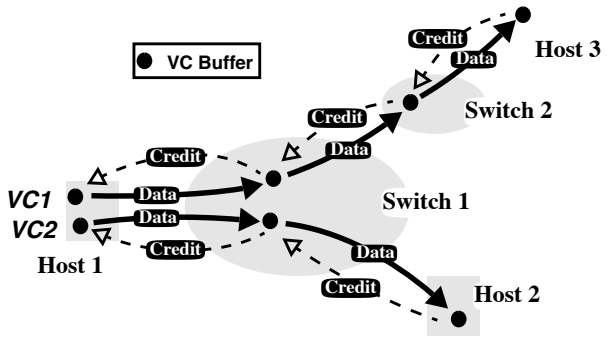


Figure 3: Credit-based flow control

maximum bandwidth of the link. More precisely, if the a VC wants to send at a rate VC_ Rate, then the buffer allocation must be at least

$$\text{Buffer\_Alloc} \geq \text{Link\_RTT} \times \text{VC\_Rate}$$

where Link_RTT is the link round trip time.

### 3.1. Credit Update Protocol (CUP)

The Credit Update Protocol (CUP) [11] is an efficient and robust protocol for implementing credit-based flow control. For each flow-controlled VC the sender keeps a running total *Tx_Cnt* of all the data cells it has transmitted, and the receiver keeps a running total *Fwd_Cnt* of all the data cells it has forwarded or dropped. The receiver will enclose the up-to-date value of *Fwd_Cnt* in each transmitted credit record for the VC. When the sender receives the credit record with value *Fwd_Cnt*, it will update the Credit_Balance for the VC by:

$$\text{Credit\_Balance} = \text{Buffer\_Allocation} - (Tx\_Cnt - Fwd\_Cnt) \quad (1)$$

where Buffer_Allocation is the total number of cells allocated to the VC. Note that the quantity, *Tx_Cnt - Fwd_Cnt*, represents the "outstanding credits". Thus the Credit_ Balance computed by Equation (1) is the proper new credit balance. [1]

The frequency that the receiver sends credit records for a VC depends on the VC's progress. More precisely, each time after the receiver has forwarded *N2* cells, the receiver will send a credit record upstream. The value of *N2* can be set either statically or dynamically. For the dynamic case, the *N2* value for a VC can be set, for example, to be one half or a quarter of the current buffer allocation for the VC [5].

### 3.2. Adaptive Buffer Allocation

Adaptive buffer allocation allows a number of VCs to share the same buffer pool dynamically. The Buffer_Allocation[2] for each VC will adapt to its current bandwidth usage. That

[1] See [11] for the use of the Credit_Check cell to recover from possible loss of data or credit cells.

is, the allocation of a VC will automatically decrease, if the VC does not have sufficient data to forward or is back-pressured because of downstream congestion. The freed up buffer space will automatically be assigned to other VCs which have data to forward and are not congested.

Adaptive buffer allocation can be implemented at the sender or receiver. As depicted by Figure 4, in a sender-oriented adaptive scheme [11], the sender dynamically allocates an shared input- buffer at the receiver among a number of VCs from the sender that share the same buffer pool. The sender allocates buffer for the VCs based on their measured bandwidth usage on the output port *p*. A previous paper [11] presents performance results of sender-oriented adaptive buffer allocation.

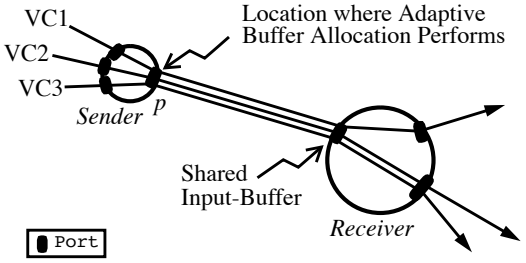


Figure 4: Sender-oriented adaptation

Receiver-oriented adaptation is depicted by Figure 5. The receiver dynamically allocates a shared output-buffer among a number of VCs from one or more senders that share the same buffer pool. The receiver allocates buffer for the VCs based on their measured bandwidth usage on the output *q*.

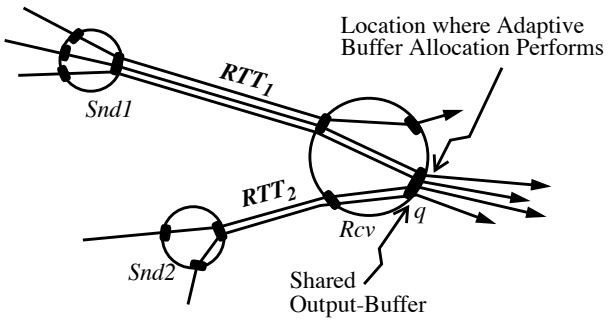


Figure 5: Receiver-oriented adaptation

Receiver-oriented adaptation is suited for the case where a common buffer pool in a receiver is shared by VCs from *multiple* upstream nodes. Figure 5 depicts such a scenario: the buffer pool at output port *q* of switch *Rcv* is shared by four VCs from two switches

[2] In the terminology of [11, 12], Buffer_Allocation would be *N2 + N3*.

*Snd1* and *Snd2*. Note that the receiver (*Rcv*) can observe the bandwidth usage of the VCs from *all* the senders (*Snd1* and *Snd2*). In contrast, each sender can only observe the bandwidth usage of those VCs going out from the same sender. Therefore, it is natural to use receiver-oriented adaptation in this case. Pseudocode for the receiver-oriented adaptation is given in [5].

Fairness is an important goal for any resource allocation strategy. In an ATM network, fairness can be measured by how two VCs share a given link that they both traverse in the same direction. Because allocation decisions must be made knowing only what can be observed at an individual switch, global fairness is generally hard to achieve. The next section gives measurement results with respect to both utilization and fairness.

## 4. Simulation Results for Credit-Based Flow Control

We have simulated the receiver-oriented credit adaptation scheme extensively against many challenging network configurations, which we call GFCs for "Generic Fairness Configurations" [5]. These configurations include the two network configurations known at the ATM Forum meeting as GFC1 and GFC2) [16]. In this section, we report some of these simulation results.

The GFC1 topology and bandwidth setup are shown in Figure 6. The link propagation delays between switches are 1800 cells (about 1000km for a 155.5 Mbps link) and 1 cell between a switch and a host. In the steady state, the expected bandwidth for each group of VCs is shown in Table 1.

Using our receiver-oriented adaptive scheme, the number of received cells on a VC at the destination is given in Figure 7. This figure gives a clear view of long term bandwidth, expressed as the slope of each line, achieved by the individual VCs. We see that bandwidth results from Figure 7 match perfectly the expected bandwidth depicted in Table 1.

| Group | Expected Bandwidth | Bottleneck Link |
|---|---|---|
| A | 1/27=0.037 | S1-S2 |
| B | 2/27=0.074 | S4-S5 |
| C | 2/9=0.222 | S3-S4 |
| D | 1/27=0.037 | S1-S2 |
| E | 2/27=0.074 | S4-S5 |
| F | 1/3=0.333 | S2-S3 |

Table 1: Expected bandwidth for GFC1

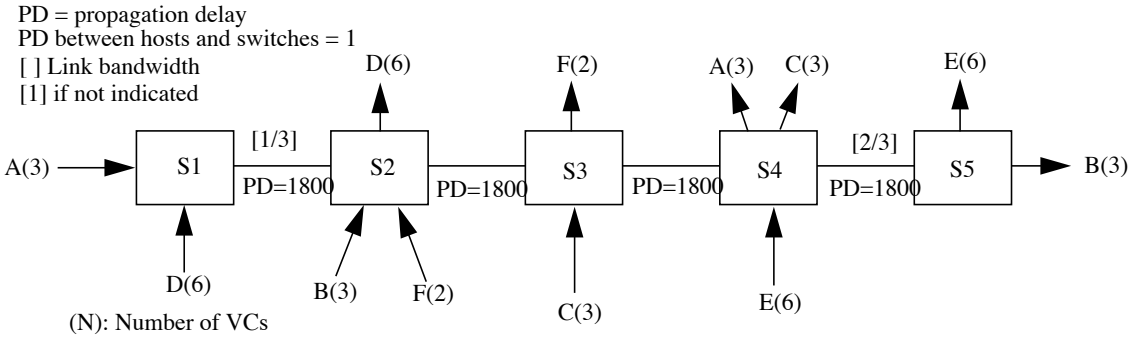


Figure 6: GFC 1 configuration

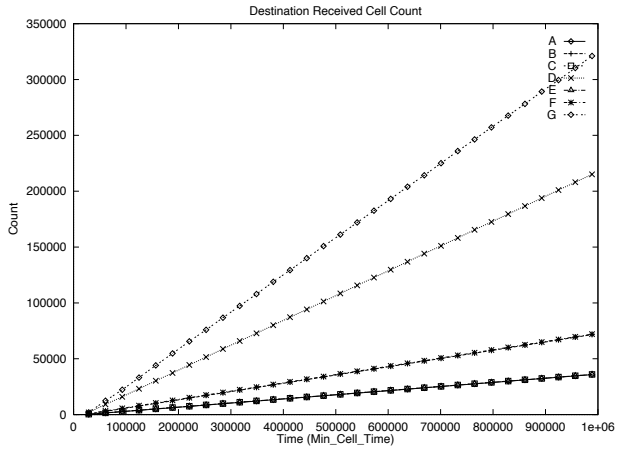


Figure 7: Destination received cell count for GFC1

GFC2, depicted in Figure 8, has a network topology similar to GFC1, but with more switches and more variety in link propagation delay.

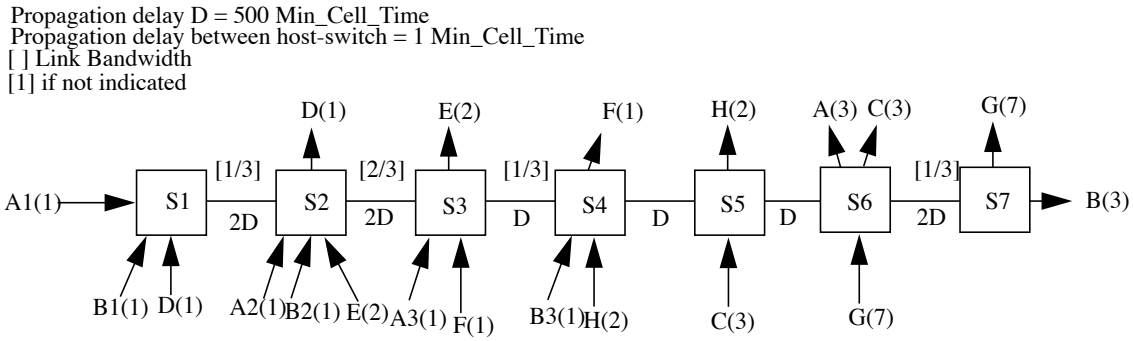


Figure 8: GFC2 configuration

The expected bandwidth of VCs in the steady state for GFC2 is shown in Table 2. Again, simulation results in Figure 9 shows that the VCs indeed achieve the expected bandwidth of Table 2.

| Group | Bandwidth | Bottleneck Link |
|---|---|---|
| A | 2/30=0.066 | S3-S4 |
| B | 1/30=0.033 | S6-S7 |
| C | 7/30=0.233 | S5-S6 |
| D | 7/30=0.233 | S1-S2 |
| E | 7/30=0.233 | S2-S3 |
| F | 2/30=0.066 | S3-S4 |
| G | 1/30=0.033 | S6-S7 |
| H | 10.5/30=0.35 | S4-S5 |

Table 2: Expected bandwidth for GFC2

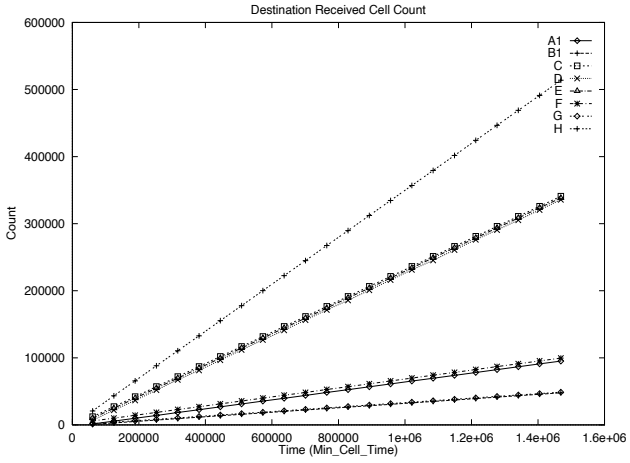


Figure 9: Destination received cell count for GFC2

GFC3 is composed of one switch and 11 hosts. The bandwidth of the input links varies from 1 to 1/100 and link propagation delay varies from 1 to 2000 Min_Cell_Time. The topology and setup are shown in Figure 10. There are 50 VCs coming in from each input port and a total of 500 VCs share the output port. In the steady state, VCs in group D and G can only reach 1/5000=0.0002 of the output bandwidth while rest of the VCs can reach 98/40000 = 0.00245.

The destination received cell count is shown in Figure 11. We see that the achieved bandwidth agrees with the expected bandwidth described in the preceding paragraph.

.

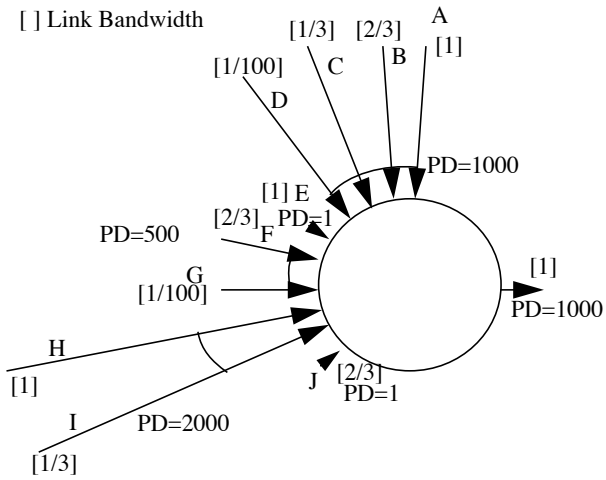


Figure 10: GFC3 configuration

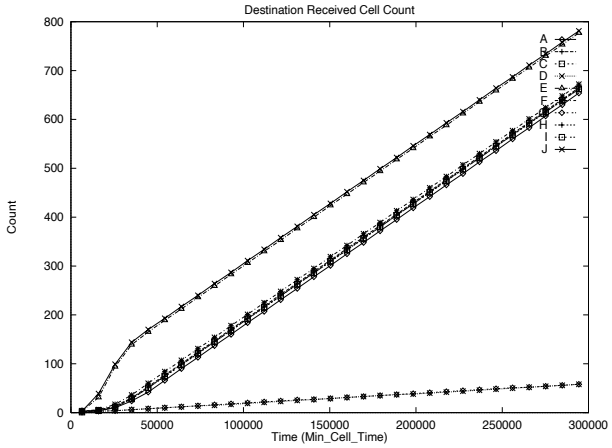


Figure 11: Destination received cell count for GFC3

In order to observe the behavior of the adaptive algorithm during severe congestion, we create a congestion situation by reducing a bottleneck link's bandwidth from 1 to 1/100 suddenly. In the following we assume the GFC3 configuration of Figure 10 with 10 VCs (one from each port) going through the switch. For the output link we start its bandwidth with 1, reduce it at 500,000 Min_Cell_Time to 1/100, and then increase it at 1,100,000 Min_Cell_Time to 1 again.

After the bandwidth reduction, those small bandwidth VCs (coming from bandwidth 1/100 link) should get fair share bandwidth, i.e., 1/1000. Figure 12 shows the fair share of bandwidth between the VCs during the bandwidth reduction period.

Moreover, Figure 12 shows that even during the bandwidth reduction period, newly started VCs can still ramp up. In particular, we see that the three VCs (C,E and H) which start at 700,000, 800,000 and 900,000 Min_Cell_Time can all ramp up.

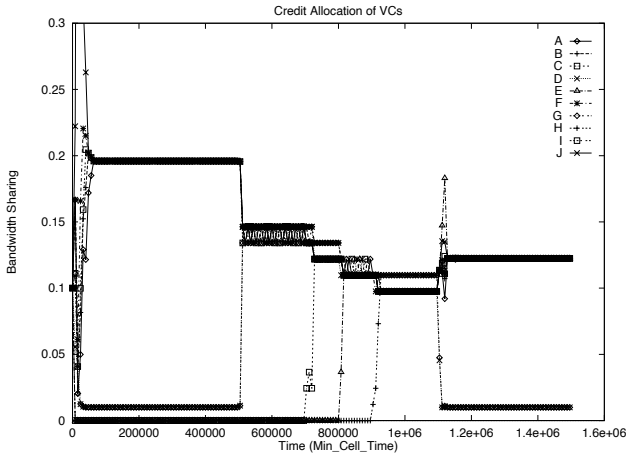


Figure 12: Bandwidth sharing among VCs for GFC3

Figure 13 shows that the three VCs achieve the same bandwidth (as their slopes are the same). Thus, during the reduced-bandwidth period, not only do new VCs can ramp up quickly, but they also get a fair share of the available bandwidth.

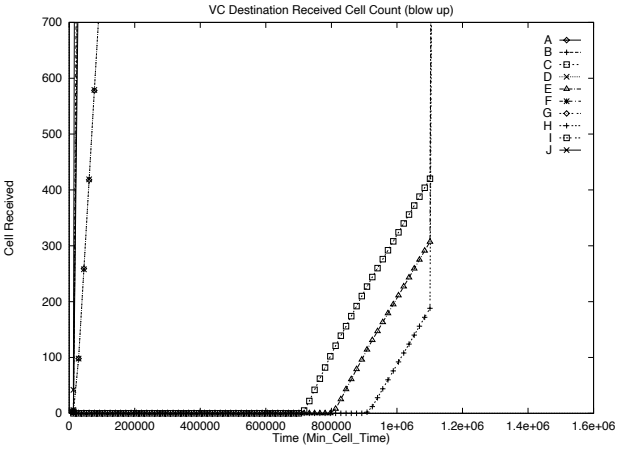


Figure 13: Destination received cell count in GFC3 for three VCs ramping up during bandwidth reduction

All of our simulation results reported in this paper assume only a memory of 4*RTT_Max*Min_Cell_Time/Output_Link_Cell_Time + 8*N cells, where N is the number of the active VCs and RTT_Max is the maximum round-trip time (in units of Min_Cell_Time) among all the input links. In fact, simulation results not reported here show that the second term can be reduced from 8*N to 2*N, by allocating a minimum buffer of two instead of eight cells, without substantial performance reduction.

## 5. Concluding Remarks

Credit-based flow control allows efficient utilization of network resources while providing high quality service to its users. This high quality services combines high peak bandwidth, zero or low loss rates and fair allocation, and does not require prearranged service contracts. We have shown in the paper that, even under stressful traffic loads and difficult network configurations, credit-based flow control with adaptive buffer allocation achieves excellent performance in terms of utilization and fairness. Using receiver-oriented adaptive buffer allocation, this performance is achieved using a moderate amount of memory. Memory requirements scale with the length of links connected to a switch, so that LAN switches can be less expensive than WAN switches.

## Acknowledgment

This research was supported in part by BNR and Intel, and in part by the Advanced Research Projects Agency (DOD) monitored by ARPA/CMO under Contract MDA972-90-C-0035 and by AFMC under Contract F19628-92-C-0116.

## References

[1] ATM Forum, "ATM User-Network Interface Specification," Version 3.0, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[2] "High-Performance Parallel Interface - Mechanical, Electrical and Signalling Protocol Specification (HIPPI-PH)", ANSI X3.183-1991.

[3] "ISDN - Core Aspects of Frame Protocol for Use with Frame Relay Bearer Service," ANSI T1.618-1991.

[4] "An Experimental Flow-Controlled Multicast ATM Switch," in these Proceedings.

[5] K. Chang and H. T. Kung, ATM_Forum/94-0929, September 1994.

[6] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Proc. SIGCOMM '89 Symposium on Communications Architectures and Protocols*, pp.1-12.

[7] H. J. Fowler and W. E. Leland, "Local Area Network Traffic Characteristics, with Implications for Broadband Network Congestion Management," *IEEE J. on Selected Areas in Commun.*, vol. 9, no. 7, pp. 1139-1149, Sep. 1991.

[8] M. W. Garrett, "Statistical Analysis of a Long Trace of Variable Bit Rate Video Traffic," Chapter IV of Ph.D. Thesis, Columbia University, 1993.

[9] V. Jacobson, "Congestion Avoidance and Control," *Proc. SIGCOMM '88 Symposium on Communications Architectures and Protocols,* Aug. 1988.

[10] H. T. Kung, "Gigabit Local Area Networks: A Systems Perspective," IEEE Communications Magazine, 30 (1992), pp. 79-89.

[11] H. T. Kung, T. Blackwell and A. Chapman, "Credit-Based Flow Control for ATM Networks: Credit Update Protocol, Adaptive Credit Allocation, and Statistical Multiplexing," *Proc. ACM SIGCOMM '94 Symposium on Communications Architectures, Protocols and Applications*, August 31-September 2, 1994, pp. 101-114.

[12] H. T. Kung and A. Chapman, "The FCVC (Flow-Controlled Virtual Channels) Proposal for ATM Networks," Version 2.0, 1993. A summary appears in *Proc. 1993 International Conf. on Network Protocols*, San Francisco, California, October 19-22, 1993, pp. 116-127. (Postscript files of this and other related papers by the authors and their colleagues are available via anonymous FTP from virtual.harvard.edu:/pub/htk.)

[13] W. E. Leland, M. S. Taqqu, W. Wilinger and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic," *Proc. SIGCOMM '93 Symposium on Communications Architectures and Protocols*, 1993.

[14] C. Ozveren, R. Simcoe, and G. Varghese, "Reliable and Efficient Hop-by-Hop Flow Control," *Proc. ACM SIGCOMM '94 Symposium on Communications Architectures, Protocols and Applications*, August 31-September 2, 1994, pp. 89-100.

[15] K.K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks," ACM Transactions on Computer Systems, Vol. 8, No. 2, pp. 158-181, May 1990.

[16] R. J. Simcoe, "Configurations for Fairness and Other Test," ATM_Forum/ 94-0557, July 1994.

[17] N. Yin and M. G. Hluchyj, "On Closed-Loop Rate Control for ATM Cell Relay Networks," submitted to IEEE Infocom 1994.

[18] C.L. Williamson and D.L. Cheriton, "Load-Loss Curves: Support for Rate-Based Congestion Control in High-Speed Datagram Networks,", *Proc. SIGCOMM '91 Symposium on Communications Architectures and Protocols*, pp.17-28.

[19]