Experimental Evaluation of ATM Congestion Control Mechanisms

Prashant Chandra, Allan Fisher, Corey Kosak, and Peter Steenkiste

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
[prashant,alf,kosak,prs]@cs.cmu.edu

Abstract

A critical issue in the design of fast packet-switch-based networks is the avoidance of data loss due to congestion. In the context of ATM networks, many link-level congestion control mechanisms for ABR traffic have been proposed and simulated, and a few have been implemented. This paper presents, for the first time, an experimental comparison of several such mechanisms. To drive this comparison, we have developed a set of benchmarks that provide a quantitative characterization of congestion control performance. We also describe a simple but novel technique, the "virtual port card", for implementing both non-native switch behavior and long link delays in ATM networks. This technique allows us to compare a wide range of mechanisms on a single flexible hardware platform. Our measurements show that while several of the ATM flow control mechanisms can eliminate cell loss, there are still several unresolved problems. First, for some traffic scenarios we see very poor application-level performance, especially in the presence of bursty traffic. Second, performance is very sensitive to the flow control parameters and identifying an appropriate set of parameters is difficult since it depends heavily on the traffic conditions.

1 Introduction

A fundamental problem in multihop switched networks is data loss due to congestion. The problem occurs when the sum of the bandwidths of the streams crossing a link exceeds the link bandwidth for an extended time period so that switch buffers fill up and overflow. The impact of congestion on end-end performance is especially bad for ATM since the loss of even a single cell often requires the retransmission of an entire Protocol Data Unit (PDU), i.e. a single dropped cell can result in the retransmission of close to 200 cells. As a result, it is important to minimize cell loss in ATM networks.

A large number of papers have been published on the topic of congestion control for ATM networks. A number of papers observe, based on simulation or measurements, that ATM cell loss due to congestion can result in extremely poor end-to-end performance. This is especially true when using the TCP/IP protocol, since its backoff policy based on timeouts can result in very low throughput in the presence of high error rates. A number of papers proposed and evaluated "localized" mechanisms to reduce the impact of cell loss. These include early packet discard [13, 24], and the use of Cell Loss Priority to indicate what packets should be dropped in the case of congestion [22]. While these localized mechanisms can reduce the impact of cell loss, it is now widely believed that closed-loop per-VC flow control is needed for ABR traffic in ATM networks.

The many ATM flow control mechanisms that have been proposed can be divided in two classes. A first class consists of roundtrip-based rate control algorithms in which a source selects a transmit rate for each VC, based on feedback from the switches along its path [4, 7, 8, 20]. Feedback can for example consist of the EFCI bit in the ATM cell header, or of an explicit rate calculated by the switches. A second class consists of hop-by-hop, credit-based flow control algorithms in which the receiver on each link provides the sender with information on buffer status for each VC. Examples include DEC AN2 [21] and Credit Net [18]. An overview of these algorithms can be found in [12, 23]. A class of rate-based algorithms has been standardized by the ATM Forum [1], although several companies are still pursuing a credit-based alternative called Quantum Flow Control [9].

In order to evaluate the different flow control algorithms in a controlled environment, the ATM Forum traffic management group developed a test suite that specifies a set of network topologies and traffic loads [2, 25]. The suite identifies a number of performance measures, including fairness, throughput, network utilization, and buffer requirements. Fairness for ABR traffic is defined as max-min fair sharing [10]. While the suite provides a solid basis for meaningful comparison, it has some shortcomings: it is intended to be used in simulations, and evaluation is limited to static traffic patterns, i.e. the effect of bursty traffic is not considered.

Most papers evaluating ATM flow control algorithms are based on simulation. The papers typically compare throughput using a specific flow control algorithm with throughput without flow control [13, 15, 17], or they focus on a class of algorithms and analyze how throughput and buffer requirements change as a function of the flow control parameters [16, 27]. Several ATM Forum contributions presented simulation results based on the test suite described above [3, 6, 26, 28]. The traffic loads are usually limited to persistent generators, although a few simulations also consider dynamic effects of bursty traffic [11]. Fewer papers have reported performance measurements of actual ATM flow control implementations. Results have been reported for credit-based flow control using DEC AN2 [29] and Credit Net hardware [19]. In both cases, the performance evaluation focused on long-running TCP connections.

In this paper we present an experimental comparison of several ATM flow control mechanisms. To drive this comparison, we have developed a set of benchmarks that provides a quantitative characterization of congestion control performance for single-bottleneck networks. The benchmarks provide insight into a wide range of flow control properties including effectiveness in reducing cell loss, fairness, responsiveness and network utilization. Our evaluation uses a simple but novel technique, the "virtual port card", for implementing both non-native switch behavior and long link delays in ATM networks. This technique allows us to compare a wide range of mechanisms on a single hardware platform.

The remainder of this paper is organized as follows. We first present in Section 2 our evaluation methodology and benchmark suite. We give a brief description of our evaluation platform, including the virtual port card, and of the flow control implementations in Sections 3 and 4. We present our measurements in Section 5 and summarize in Section 6.

2 Evaluating Congestion Control Mechanisms

Our quantitative evaluation of multiple congestion control mechanisms focusses on single-congested-link scenarios, making a comprehensive evaluation possible. In this section we first outline the more general problem of evaluating flow control mechanisms before describing in detail the specific configurations that we use in our evaluation.

2.1 Properties of congestion control mechanisms

Congestion control algorithms can be evaluated by measuring network performance properties that are (in part) determined by the congestion control, and by direction measurement of flow control properties. We describe both classes of properties.

Application level performance Applications typically evaluate network performance based on the time it takes to transfer their messages. Depending on the size of the message, this will require low packet latency (small messages) or high sustained throughput (large messages). A good congestion control algorithm performs well for a number of types of traffic, so it can provide good service to, for example, both high-bandwidth and low-latency applications. This requires a flow control mechanism that is highly responsive and has low data loss.

Network utilization Network resources must be utilized efficiently. This means that, even under dynamic network conditions, the congestion control mechanism should prevent under-utilization of network links, and that efficient link utilization should be achieved with minimal resources on switches, e.g. small buffer sizes. This requires a flow control algorithm with low data loss, high responsiveness, and low overhead.

Fairness A good congestion control algorithm should allocate network resources in a fair manner amongst competing connections. The specific fairness criterium that has been defined for ATM ABR traffic is fair sharing. Fairness has both a static and dynamic component. The static fairness requirement is that, without changes in the traffic load, the network converges to a state where competing connections share bandwidth fairly. Under dynamic conditions, short term unfairness is often unavoidable, but the system should converge quickly to the new fair state. Fairness requires a congestion control algorithm that both converges correctly and is responsive.

Achieving good network performance requires a flow control mechanism with the following characteristics:

Data Loss High data losses reduce application performance (decrease sustainable throughput and increase average latency) and network utilization. Congestion control mechanisms should be designed to minimize data loss under the most severe congestion conditions and prevent congestion collapse of the network.

Responsiveness The traffic load placed on a network changes continuously and poor responsiveness to these changes reduces application performance, network utilization and fairness. A good congestion control algorithm should be responsive to dynamically changing conditions in the network: sources should quickly throttle back their data rate on the onset of congestion and should be able to quickly ramp up to any additional bandwidth that may become available in the network. The responsiveness of the congestion control algorithm depends on the delay in the feedback loop.

Overhead A good congestion control algorithm has very little overhead: the network bandwidth required for control messages is small and the processing requirements at the network nodes and end-systems are low. High overhead reduces network utilization.

The congestion control algorithm features listed above are mainly performance oriented. In addition, there are also a number of important implementation-oriented features. First, a good congestion control algorithm should be *robust* against loss of feedback information from the network. Loss of control messages should not cause congestive collapse of the network. Second, a congestion control mechanism must be ultimately *implementable*. Finally, congestion control algorithms should be *scalable*, i.e. they should perform well with varying speeds of links in the network and scale to a large number of user connections. Note that while it seems desirable to have a single algorithm for LANs, MANs and WANs, an algorithm that performs well for one category might be acceptable.

2.2 Approach

Our evaluation of congestion control methods focusses on the performance-oriented features since they are easiest to measure quantitatively. Properties such as robustness and implementation complexity are harder to characterize in a quantitative manner.

A quantitative analysis involves defining what network *configurations* are used in the measurement, and defining *traffic models* that capture the load that is placed on the network configuration during the measurements. For each combination of configurations and traffic models we also have to identify what *metrics* should be used, i.e. what quantities should be measured. A wide range of choices exists for each of these three variables:

Configurations The configurations used should be representative of actual configurations found in real networks. Some configurations that have been proposed are peer-to-peer configuration, fairness configurations (GFC 1 and GFC 2), ring configurations, and hierarchical configurations (found in telephone networks).

Traffic models The traffic models used should be representative of actual network traffic. Examples include persistent traffic (high-bandwidth), open loop bursty (various stochastic modes such as ON/OFF, Poisson etc.), and closed loop bursty (request/response or transaction-oriented traffic).

Metrics We can distinguish between *end-system-oriented metrics* such as packet loss, throughput, and response time, and *network-oriented metrics* such as buffer requirements in the switches, network utilization, (long-term and short-term) fairness, and overhead of control messages.

The evaluation of a specific set of congestion control algorithm properties requires the selection of a suite of tests, each of which is a suitable combination of configurations, traffic models and metrics.

2.3 A test suite for single-bottleneck topologies

The test suite given here combines a suitable set of metrics, traffic models and configurations, to evaluate single-congested-link scenarios in a quantitative way. The configuration used is a peer-to-peer configuration shown in Figure 1. It consists of a switch and three nodes representing end-points, S_{far}, S_{near}, and D. The link between the switch and D is the congested link.

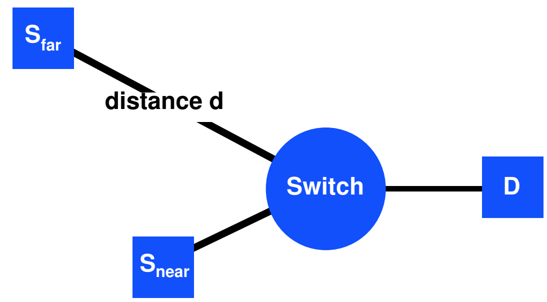

Figure 1: Test configuration

Experiment 1: Static rate matching This experiment uses only nodes S_{far} and D. S_{far} is a persistent packet source, i.e. it continuously sends packets at line rate. The output link (to node D) is run at a lower rate than the input link (from node S_{far}), and the congestion control mechanism is used to provide rate-matching. We vary switch buffer sizes and link distances, and measure data loss and throughput. This experiment provides insight into the efficiency, stability and resource requirements of the congestion control mechanisms.

Experiment 2: Burst bandwidth allocation Again, only nodes S_{far} and D are used. Node S_{far} sends packets of varying lengths, and receives short acknowledgements from D. The delay of the link is varied, and the request-response latency is measured for a variety of request sizes. This experiment shows how effectively the congestion control mechanisms can respond to bursty demands.

Experiment 3: Static congestion In this experiment, both the nodes S_{far} and S_{near} are persistent packet sources. The delay of the input link of S_{far} is varied between LAN and WAN dimensions. We measure data loss and throughput of each stream, as well as utilization of the link to node D, as switch buffer sizes are varied. This experiment sheds light on data integrity, efficiency, scalability and long-term fairness of the congestion control method under consideration.

Experiment 4: Dynamic congestion In this experiment, node S_{far} is a persistent source and node S_{near} is a ON-OFF source. S_{far} is connected via a WAN link to the switch SW. The ON-OFF duty cycle of S_{near} is varied. We measure utilization of the link to D and the throughput and loss rate of S_{far}. This provides insight into the responsiveness of the congestion control algorithm, and hence its efficiency under dynamic conditions.

Experiment 5: Dynamic fairness In this experiment, node S_{far} is again connected via a long link to SW. Both S_{far} and S_{near} are request/response sources and they query a server D. Requests have a fixed size and the inter-request times are exponentially distributed. The load generated by the sources is varied and the number of requests received at node D from both sources is measured. This experiment gives insight into throughput and fairness under dynamic conditions.

We believe that this set of experiments provides a comprehensive picture of congestion control mechanisms for single-congested-link scenarios. It does not address scalability properties with respect to number of nodes/users and fairness properties with multiple bottleneck scenarios (GFC configurations).

3 Experimental Infrastructure

Experiments were run on Credit Net, an experimental ATM local area network. Credit Net incorporates switches, host interfaces, and OC-3 and OC-12 SONET links. The Credit Net switch hardware [19] provides native support for per-VC buffering and feedback of absolute buffer credits. The Credit Net adapter [14] uses DMA to send and receive ATM data on a host's PCI I/O bus. It performs SAR and AAL5 functions on behalf of the host, and manages multiple queues of buffers for send and receive. The adapter also provides a variety of mechanisms for implementing congestion control algorithms:

- The adapter incorporates a microcode engine that can control the adapter's transmit scheduler, manipulate adapter memory, and send and receive cells on the network link.
- The adapter can be augmented with an embedded i960 microprocessor that can manipulate adapter and host memory, and coordinate with adapter microcode.
- The adapter provides the host processor with extensive access to its internal data structures, and with means to coordinate with the adapter's microcontroller.

These facilities allow implementation of a variety of endpoint behaviors with high performance. Furthermore, we can use the adapter in a "virtual port card" configuration to provide switch congestion control functionality beyond that provided in switch hardware. As is shown in Figure 2, a host with two network adapters (NA) provides the input port, output port, and scheduling capabilities of a single port of a switch.

The key difference between a virtual port and a real port is that the virtual port is downstream from the "merge" point in the switch, where we expect congestion resolution to take place. This may create the problem that cell loss occurs in the switch, away from the congestion control mechanism running on the virtual port. We avoid this condition by scheduling the virtual port's outgoing link to run at a slower rate

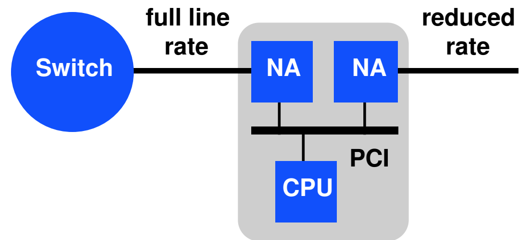

Figure 2: Virtual port card configuration

than its incoming link, moving the point of the congestion to the virtual port. While in a completely uncontrolled situation, large congestion losses could still occur at the switch, this should not happen since we are testing mechanisms designed to control congestion, and we expect to see little or no loss of data at the switch. Monitoring of data loss at the switch during experimental runs confirms this expectation.

The virtual port card can also model the behavior of a long link. This is accomplished simply by feeding the scheduled output of the virtual port card into a delay FIFO of constant length, with its output rate matched to the link rate by dropping incoming cells when they arrive faster than the outgoing link can accommodate them, and inserting null cells when incoming cells arrive at less than the outgoing link rate.

4 Congestion Control Implementations

We briefly describes the congestion control implementations used in our experiments. We describe any parameter settings adopted, any simplifications or modifications of the basic algorithms, and our mapping of the mechanisms onto the Credit Net hardware.

Credit with static buffering This method uses the native flow control capabilities of the Credit Net LAN switch, which implements the FCVC N123 algorithm [18]. The N123 algorithm is implemented at the host, for these tests, on the embedded i960 processor described above. Buffer space on the switch is allocated in proportion to the propagation delay of the upstream link, in order to eliminate data loss. Because of limits on buffer space in the switch, we use the virtual port card implementation for long-latency links.

Credit with adaptive buffering This mechanism uses the same host implementation as the previous case; the only change is at the switch. Instead of making a worst-case allocation of buffer space per VC, the switch allocates space to each VC in proportion to its bandwidth, adjusting allocations periodically to track bandwidth changes. We implement adaptive buffering using the virtual port card approach; the host processor of the virtual port card runs the CUP algorithm and performs periodic buffer reallocation [15]. It is similar to the QFC algorithm [9].

Rate control with EFCI This case and the following one implement two variants of the ATM Forum ABR congestion control mechanism[1, 4]. To make implementation easier, we have simplified the format of the resource management cell, and we have not implemented timeout-related behaviors that do not arise in our test cases. The endpoint behaviors are implemented using a combination of the microcode and embedded processor cabilities of the host adapter [5]. We have implemented a rate scheduler, capable of scheduling outgoing traffic and resource management cells at OC-3 line rates, in the adapter's microcode. The embedded processor at the source is responsible for generating outgoing resource management (RM) cells, for receiving and interpreting incoming resource management cells, and for updating the transmission rates used by the scheduler.

The EFCI (explicit forward congestion indication) version of ABR algorithm calls for an additive rate increase when returning RM cells indicate no congestion, and a multiplicative decrease when returning RM cells indicate congestion. The embedded processor at the destination is responsible for receiving forward RM cells, inserting congestion indications based on the payload type congestion indications of recent incoming cells, and reflecting them back toward the source.

The switch behavior of the EFCI mechanism, implemented on the virtual port card host processor, is simply to add a congestion indication to the headers of cells that depart on a port whose buffer is filled beyond a preset threshold.

Rate control using EPRCA The Enhanced Proportional Rate Control Algorithm (EPRCA) is a variant of the ATM Forum ABR scheme in which switches provide sources with explicit rate bounds. Endpoint behavior is implemented as in the previous case, except that the source updates rates to the value provided in the explicit rate field of returning RM cells. Switch behavior is more complicated: the switch must intercept forward RM cells and periodically use their current rate information to compute a fair share rate. It must also set the explicit rate field in reverse RM cells to the minimum of the incoming rate and the fair share rate. The fair share rate is computed using the exponential averaging algorithm specified by the ATM Forum.

5 Measurements

In this section we present and analyze our performance measurements for the different congestion control implementations. All end-points and virtual ports consisted of 133 MHz and 166 MHz Pentium PCs running NetBSD.

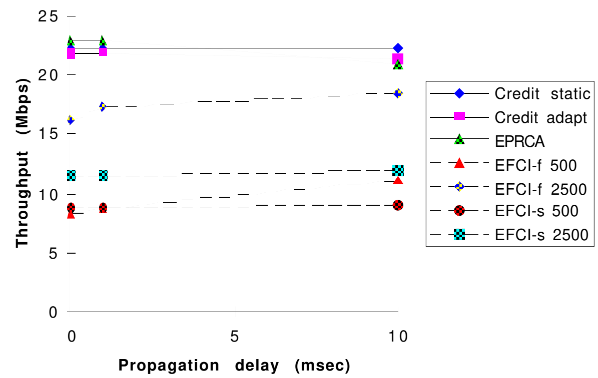

Figure 3: Throughput as a function of link delay

Static rate matching First, we consider the behavior of the congestion control mechanisms in Experiment 1, dealing with rate matching of a static stream traversing a faster link feeding into slower link. The slow link is throttled to 1/6 of OC-3 bandwidth, and the source attempts to send at 70 Mbps subject to congestion control. Figure 3 shows how the throughput of valid packet data, for IP MTU-sized packets, depends on the propagation delay of the fast link. The cases plotted include static and adaptive credit-based flow control, EFCI rate control with a variety of rate adaptation parameters, and EPRCA rate control. For EFCI, "-f" (fast) corresponds to a rate increase of 10 Mbps and a congestion slowdown of 50%; "-s" (slow) corresponds to an an increase of 100 Kbps and a decrease factor of 10%. The number following -f or -s is the congestion threshold in cells; all cases use a buffer that holds 4000 cells.

As one would expect, credit with static buffering gives the best performance, although both adaptive credit-based flow control and EPRCA give virtually identical performance. The reason is that with static traffic sources, the latter two algorithms stabilize at a point with high network utilization. Rate-based flow control using EFCI gives lower performance as a result of oscillations. For EFCI, throughput is best using an EFCI/drop threshold of 2500/4000 bytes and aggressive parameter settings. Note however that there is a cell loss of about 20% using the 2000/4000 threshold for the "fast" rate adaptation parameters, so with a reliable transport protocol added, this case is likely to give poor application level performance. In other words, for this experiment, rate control using EFCI performs best using conservative parameter settings.

Burst bandwidth allocation We now consider Experiment 2, which measures the response time of a transaction consisting of the transfer of a block of data across a link, and an acknowledgement in the reverse direction. This experiment characterizes the speed with which burst bandwidth is allocated. We measure response time for two link latencies (LAN and WAN) and, in the EFCI case, for several settings of

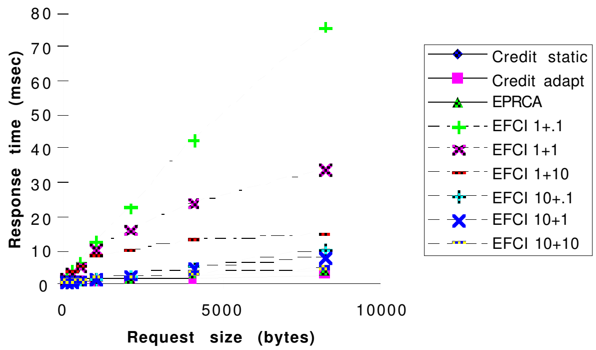

Figure 4: Response time as a function of request size (LAN)

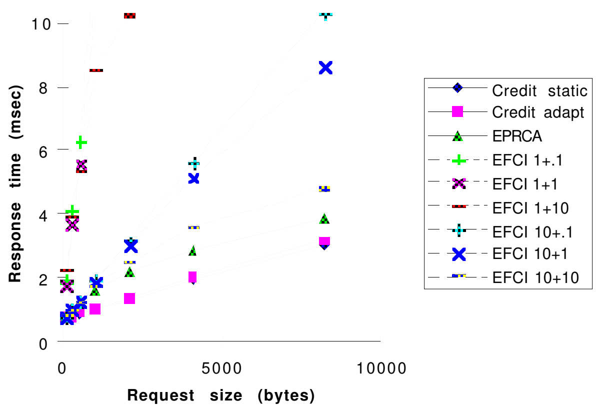

Figure 5: Response time as a function of request size (LAN)(magnified scale)

the rate adaptation parameters. Requests are sent at 70 Mbps, and the ABR implementations are allowed to settle to their initial cell rates between requests.

Figure 4 shows response time as a function of request size in a LAN with negligible propagation delay. The EFCI curves are marked INIT+INCR, where INIT is the initial rate in Mbps and INCR is the rate increase in Mbps applied each time a resource management cell without a congestion indication is received. The EPRCA initial rate is 10 Mbps. Because this scenario does not induce congestion, rates are never lowered except when the stream is idle. Figure 5 shows the same curves on a magnified response time scale.

In this graph, the static credit algorithm provides the best response time, since its worst-case buffer allocation requires no ramp-up delay. The dynamic credit algorithm has similar performance, because our implementation of CUP uses a coarse grain buffer allocation scheme, so there is sufficient buffer space available to send one packet without waiting for credit. The EFCI response times are dominated by the initial data rate for short requests, since the entire request may be completed before any returning RM cells are received. For longer packets and lower initial rates, returning RM cells arrive before the end of transmission, so the incremental rate increase has an impact on performance. EPRCA performs slightly better than EFCI on longer requests, since its first returning RM cell permits it to ramp up to full bandwidth immediately.

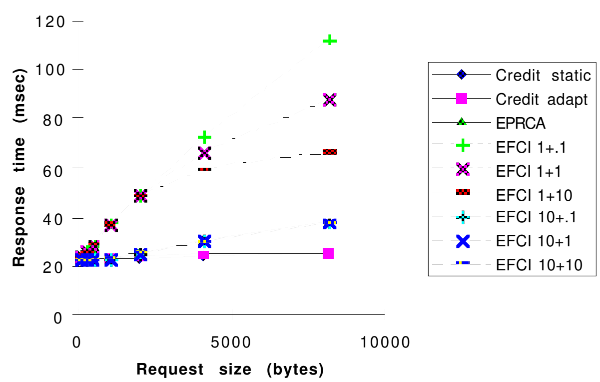

Figure 6: Response time as a function of request size (WAN)

Figure 6 shows response time using a link with a propagation delay of 10 msec. All response times in this graph have an offset of 20 msec, corresponding to the physical round-trip time of the long link. In this graph, the static and adaptive credit algorithms again provide the best response time. The reason is again the lack of ramp-up delay. Because of the long delay before RM cells return, initial rate is an even more important factor for the EFCI scheme than before. For the same reason, EPRCA performs about the same as EFCI for an initial rate of 10 Mbps.

Overall for this experiment, rate control using EFCI performs best using the more aggressive parameter settings. The reason is that these settings allow the algorithm to adapt more quickly and oscillations, which caused problems in the static rate matching experiment, are not a concern because the data transfer is very short.

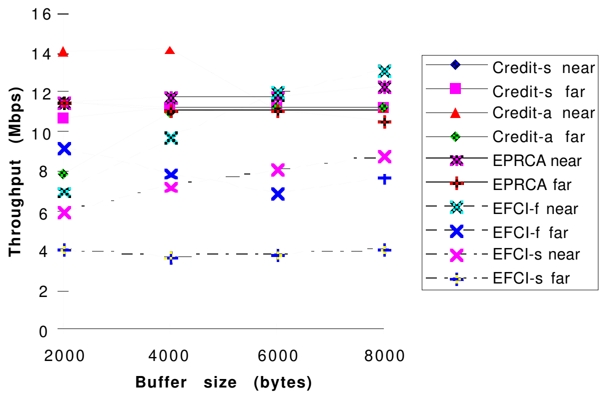

Figure 7: Throughput as a function of buffer size under static congestion

Static congestion This experiment creates congestion between a stream traversing a long link and one traversing a short link. All results are for a bottleneck link bandwidth of 25 Mbps. Figure 7 shows sender packet data throughput for this scenario as a function of switch buffer size allocated to the high-latency connection. In this case, the throughput reflects both loss of efficiency, unfairness and loss of data due to the congestion control scheme used.

The EPRCA and credit algorithms exhibit both high link utilization and fairness, although the fairness of EPRCA degrades slightly as buffer size increases. There is also negligible cell loss using these algorithms. The results using the EFCI algorithm exhibit unfairness. The more conservative parameter setting result in better fairness, since there is less oscillation. Moreover, there is a cell loss ratio of about 7% for the fast parameter settings, while there is no cell loss for the more conservative EFCI parameter settings, making the conservative settings the better choice for this experiment.

Dynamic congestion This experiment examines the behavior of a persistent stream, traversing a long link, as it encounters congestion created by a local ON-OFF source. The persistent stream crosses a link with a delay of 10 ms, and the congested link has a bandwidth of 25 Mbps. Figure 8 shows how the persistent stream's sender-side throughput of packet data, for IP MTU-sized packets, depends on the average offered bandwidth of the bursty source. The credit and EPRCA algorithms perform best, since they stabilize the rates at a fair rate, similar to what we observed in the previous experiment. Both settings of the EFCI algorithms give poor performance relative to the credit and EPRCA algorithms as a result of oscillations. While it appears that the aggressive parameter settings result in higher throughput, they also result in a cell loss rate of about 20%, while there is no cell loss with conservative settings, so we expect the conservative settings to results in better application level throughput.

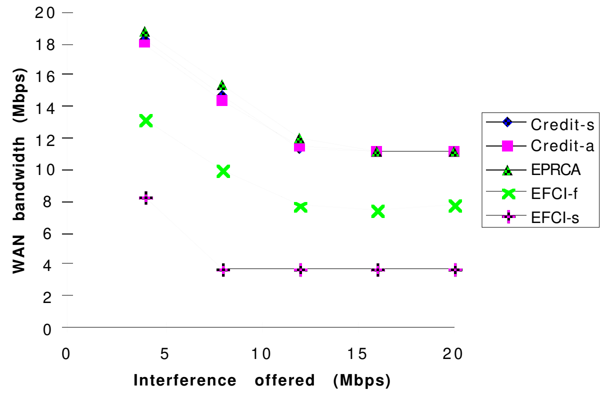

Figure 8: Throughput of a persistent WAN source with bursty interference

Notice that for each of the algorithms, throughput initially drops as the interfering load increases, but it reaches a minimum value and remains constant, insensitive to further increases of the competing load. This is a direct result of the flow control mechanisms, which limit the bursty interference source to a certain rate. Increasing the rate of the interfering source above this rate will not have any effect on how much data will be placed on the wire by the bursty source, so the throughput of the persistent source stabilizes. Moreover, at these high rates the bursty source becomes a persistent source so we would expect rates corresponding to the flat segments in Figure 8 to be similar to the results of the previous experiments; a comparison of Figures 7 and 8 confirms this. Finally, we would expect the flow control algorithms that achieve the lowest utilization of the bottleneck link to reach this point first (i.e. for the lowest competing load), which Figure 8 confirms.

Dynamic fairness The final experimental configuration runs two request-response streams, one entering a congested link from a long-delay link (10ms) and the other from a short-delay link. The bottleneck link bandwidth is 25 Mbps and the request size is 2 KByte. Each source generates requests in a Poisson process, and the load is increased by reducing the average interpacket time. The rate configurations use a 4000-cell buffer with a congestion threshold of 2000 cells, an initial rate of 10 Mbps, an increase rate of 100 kbps, and a congestion reduction factor of 10%.

Figure 9 shows throughput for each stream as a function of the load offered. The credit and EPRCA configurations manage this situation well, up to and

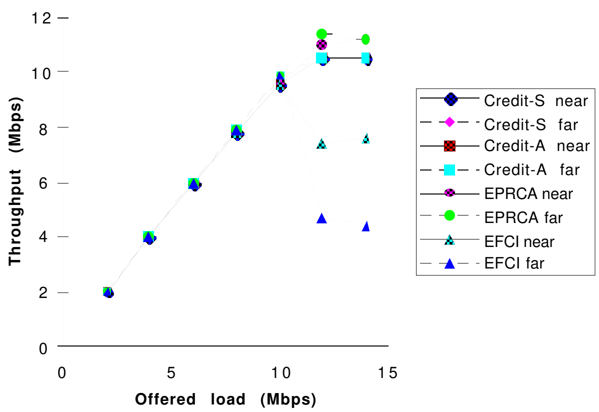

Figure 9: Requests received as a function of request size

beyond the point of link saturation. In contrast, as the EFCI implementation approaches 80% link utilization, its performance degrades substantially due to oscillations. High loads in this scenario also result in unfair allocation of bandwidth between the local and wide-area sources. Note that under high loads, the bursty sources turn into persistent sources as a result of the backpressure exercised by the flow control algorithms.

Acknowledgment

This research was sponsored by the Defense Advanced Research Projects Agency under contract number F19628-92-C-0116.

We thank our colleagues at Intel Architecture Laboratories for their vital contributions. We also thank past and present Credit Net contributors at CMU, Harvard, and BNR for many useful discussions.

6 Conclusion

This paper presents a comprehensive experimental evaluation of a variety of mechanisms for congestion control in a single-congested-link scenario. Our measurements show that several of the ATM flow control mechanisms can eliminate cell loss, but that there are a number of unresolved problems.

First, under some traffic conditions we sometimes see very poor application-level performance. Our results show that conservative credit gives the best results. This is not a surprise since it in effect reserves sufficient resources to achieve high performance. The adaptive credit and EPRCA algorithms behave in very similar ways and give good performance in all static tests. In the bursty tests they sometimes result in slow response times and unfairness as a result of slow ramp-up. The slow ramp-up is a fundamental problem: sending data at high rates without or with data loss either requires the presence of sufficient resources in the network (such as enough spare bandwidth or buffer space), or requires the sender to wait for feedback from the network, which result in slow ramp-up. Rate based flow control based on EFCI performs poorly in all tests. The poor performance is caused by oscillation which is a fundamental part of the algorithm.

The second problem area is parameter selection. Although operating parameters for adaptive control algorithms can be tuned to individual situations, no parameter setting is best under all conditions. Appropriate parameters depend heavily on traffic conditions, and defining algorithms for selecting flow control parameter settings for specific connection that will provide good application level performance, including both low cell loss and good response times, is an important open research area.

References

[1] ATM Forum Technical Committee. Traffic management specification version 4.0, April 1996.

[2] Andrew W. Barnhart. Baseline model for rate-control simulation. ATM Forum, Traffic Management Working Group submission 94-0399, Hughes Network System, May 1994.

[3] Jon C. R. Bennett, Koling Chang, H. T. Kung, and Dong Lin. A comparison of EPRCA September 1994 version and FCVC control schemes: Simulation results. ATM Forum, Traffic Management Working Group submission 94-0929, Fore Systems, Harvard University and Sandia National Laboratories, September 1994.

[4] Flavio Bonomi and Kerry Fendick. The rate-based flow control framework for the available bit rate ATM service. *IEEE Network Magazine*, 9(2):25–39, March/April 1995.

[5] Prashant Chandra, Allan Fisher, Corey Kosak, and Peter Steenkiste. Implementation of ATM endpoint congestion control protocols. In *Proceedings of the IEEE Hot Interconnect Symposium '96*, Palo Alto, CA, August 1996. IEEE.

[6] Koling Chang and H. T. Kung. Performance of credit-based flow control on challenge configuration. ATM Forum, Traffic Management Working Group submission 94-0880, Harvard University and Sandia National Laboratories, September 1994.

[7] Anna Charny and K. K. Ramakrishnan. Time scale analysis of explicit rate allocation in ATM networks. In *Proceedings of INFOCOMM 96*, March 1996.

[8] Anna Charny, K.K. Ramakrishnan, and Anthony Lauck. Scalability issues for distributed explicit rate allocation in ATM. In *Proceedings of INFOCOMM 96*, March 1996.

[9] Mike Gaddis and Walter Kelt (eds). Quantum flow control. Technical Report FCC-SPEC-95-1, Flow Control Consortium, July 1995.

[10] Ellen L. Hahne. Round-robin scheduling for max-min fairness in data networks. *IEEE Journal on Selected Areas in Communication*, 9(7), September 1991.

[11] David Hughes and Pat Daley. More ABR simulation results. ATM Forum, Traffic Management Working Group submission 94-0777, StrataCom, Inc., September 1994.

[12] Raj Jain. Congestion control and traffic management in ATM networks: Recent advances and a survey. *Computer Networks and ISDN Systems*, February 1995.

[13] Lampros Kalampoukas. Performance of TCP over multi-hop ATM networks: A comparative study of ATM-layer congestion control schemes. Technical Report 95-13, University of California, Santa Cruz, February 1995.

[14] Corey Kosak, David Eckhardt, Todd Mummert, Peter Steenkiste, and Allan Fisher. Buffer management and flow control in the credit net ATM host interface. In *Proceedings of the 20th Conference on Local Computer Networks*, pages 370–378, Minneapolis, October 1995. IEEE.

[15] H. T. Kung, T. Blackwell, and A. Chapman. Credit update protocol for flow-controlled ATM networks: Statistical multiplexing and adaptive credit allocation. In *Proceedings of the SIGCOMM '94 Symposium on Communications Architectures and Protocols*, pages 101–114. ACM, August 1994.

[16] H. T. Kung and Koling Chang. Receiver-oriented adaptive buffer allocation in credit-based flow control for ATM networks. In *IEEE INFOCOM '95*, volume 1, pages 239–252, Boston, Massachusetts, April 1995. IEEE.

[17] H. T. Kung, Robert Morris, Thomas Charuhas, and Dong Lin. Use of link-by-link flow control in maximizing ATM performance: Simulation results. In *Proceedings of the IEEE Hot Interconnect Symposium '93*, Palo Alto, CA, August 1993. IEEE.

[18] H.T. Kung and A. Chapman. The FCVC (flow controlled virtual channels) proposal for ATM networks. In *International Conference on Network Protocols*, San Francisco, CA, Oct 1993. IEEE.

[19] Robert Morris and H. T. Kung. Impact of ATM switching and flow control on TCP performance: Measurements on an experimental switch. In *GLOBECOM '95*. IEEE, November 1995.

[20] Hiroyuki Ohsaki, Masayuki Murata, Hiroshi Suzuki, Chinatsu Ikeda, and Hideo Miyahara. Rate-based congestion control for ATM networks. *Computer Communication Review*, April 1995.

[21] Cüneyt Özveren, Robert Simcoe, and George Varghese. Reliable and efficient hop-by-hop flow control. In *Proceedings of the SIGCOMM '94 Symposium on Communications Architectures and Protocols*, pages 89–100, University College, London, London, UK, October 1994. ACM.

[22] Michael Perloff and Kurt Reiss. Improvements to TCP performance in high-speed ATM networks. *CACM*, 38(2), February 1995.

[23] K. K. Ramakrishnan and Peter Neuman. Integration of rate and credit schemes for ATM flow control. *IEEE Network Magazine*, 9(2):49–56, March/April 1995.

[24] Allyn Romanow and Sally Floyd. Dynamics of TCP traffic over ATM networks. In *Proceedings of the SIGCOMM '94 Symposium on Communications Architectures and Protocols*, pages 79–88, University College, London, London, UK, October 1994. ACM.

[25] Robert J. Simcoe. Test configurations for fairness and other tests. ATM Forum, Traffic Management Working Group submission 94-0557, Digital Equipment Corporation, July 1994.

[26] Kai-Yeung Siu and Hong-Yi Tzeng. Adaptive proportional rate control (APRC) with intelligent congestion indication. ATM Forum, Traffic Management Working Group submission 94-0888, University of California, Irvine, September 1994.

[27] Kai-Yeung Siu and Hong-Yi Tzeng. Intelligent congestion control for ABR service in ATM networks. *Computer Communication Review*, 1995.

[28] Hong-Yi Tzeng and Kai-Yeung Siu. Comparison of performance among existing rate control schemes. ATM Forum, Traffic Management Working Group submission 94-1078, University of California, Irvine, November 1994.

[29] Hongbo Zhu, Luiz A. Dasilva, Joseph B. Evans, and Victor S. Frost. Performance evaluation of congestion control mechanisms in ATM networks. In *Computer Measurement Group Annual Conference (CMG '95)*, December 1995.