FAIRNESS AND CONGESTION CONTROL ON A LARGE ATM DATA NETWORK WITH DYNAMICALLY ADJUSTABLE WINDOWS

E. L. HAHNE, C. R. KALMANEK, and S. P. MORGAN

AT&T Bell Laboratories
Murray Hill, New Jersey 07974-2070, USA.

We propose procedures for dynamically adjusting window size on a high-speed, wide-area virtual-circuit data network using sliding-window flow control, so as to use buffer space at the switching nodes efficiently while giving each user a fair share of the network bandwidth under both light traffic and heavy traffic conditions. The network uses round robin queueing disciplines at routers and switches, and provides signaling protocols whereby users who follow the protocols will never lose data because of buffer overflow within the network. Simulations verify that the network protocols work as expected. If the end users employ independent retransmission protocols, some communication between network and users about windows may be needed in order to achieve optimal performance.

1. INTRODUCTION

This paper addresses the problems of fairness and congestion control on a high-speed, wide-area ATM network. We restrict our attention to asynchronous data traffic, inasmuch as the bursty nature of data traffic generally makes it harder to handle than synchronous traffic. A service discipline that carries both synchronous and asynchronous traffic within the spirit of the present approach has been described in [1].

In Fig. 1, hosts $H_A$ and $H_B$ are connected to local area networks $L_A$ and $L_B$ that are connected through routers $R_A$ and $R_B$ to the switches of a wide-area virtual-circuit network. Each router accepts packets from its local network that are addressed to other local networks, divides the packets into ATM cells, and transmits the cells to the virtual-circuit network. The exit router reassembles the cells into packets for its local network. The virtual-circuit network may operate at speeds in the range from 45 Mb/s to 600 Mb/s. The hosts may employ connectionless protocols independent of the protocols used by the virtual-circuit network.

Our work has three objectives, namely fairness, stability, and efficiency. Fairness means that well-behaved users are not harmed by abusive users, and that under congestion the network divides its resources in a controlled way. Stability means that as the offered load increases without limit, the useful throughput of the network remains near its theoretical maximum. Efficiency means that network bandwidth and buffer space are not wasted. The full bandwidth of the network is made available quickly to those who are able to use it, and buffer space requirements at queueing points are not excessive.

Fairness is achieved by serving virtual circuits according to some type of round robin discipline, rather than first-in-first-out. Under moderate loads, round robin disciplines give short delays to short messages. Furthermore, as discussed by Morgan [2] and others, round robin disciplines are fair under heavy-traffic conditions. When there is not enough bandwidth to go around, round robin disciplines divide the available bandwidth equally, so that light users are not locked out by users with large demands. We use round-robin service at switches, and a similar discipline [3] at routers.

Stability of high-speed wide-area data networks under congestion is a more difficult problem. Suppose that a sliding-window protocol is used to flow-control each virtual circuit. Let the window
size, that is, the maximum amount of data that the virtual circuit is allowed to have in transit at any given time, be $W$. If the transmitter and receiver are connected by a transmission path of speed $S$ with a round-trip propagation time $T_o$, then in order to maintain continuous transmission on an otherwise idle path, $W$ must be at least as large as the full-speed round-trip window $W_o = ST_o$. If $W$ is less than $W_o$, then the average fraction of the network bandwidth that the circuit gets cannot exceed $W/W_o$.

In principle, if a virtual circuit has a window of a given size and buffer overflow is to be rigorously avoided, buffer space adequate to store the entire window must be available at every queueing point. On a lightly loaded network a circuit usually does not occupy much buffer space, and there can be statistical sharing of buffers among different users with negligible probability of overflow. On a congested network the probability of shared-buffer overflow is no longer negligible. Losses due to overflow trigger retransmissions, which can lead to instability.

Various methods of avoiding congestion instability have been proposed. One approach \cite{4,5} is to have users voluntarily reduce their window size at the onset of congestion. A popular alternative to window flow control is to require users to control their transmission rates, and to guarantee to carry a user's traffic only so long as the user's long-term and short-term average rates do not exceed fixed values \cite{6-9}. It is not yet clear \cite{10} whether rate-control schemes can work satisfactorily with computer traffic, which is intrinsically highly bursty.

Our study starts from the well-known fact that congestion instability due to data loss does not occur in a virtual-circuit network if windows are used and a full window of memory is preallocated to each virtual circuit at each queueing node. If preallocated buffers of size $W_o$ are combined with round-robin queueing, the network is stable and as fair as it can be under the given load. It may be practicable to dedicate this much memory to each virtual circuit at each node of a small or low-speed network \cite{11}. However, as network speeds and sizes increase, it ultimately ceases to be feasible to supply a buffer of size $W_o$ for every virtual circuit. For example, for a transcontinental network with a round-trip time of 60 ms, at a speed of 1.5 Mb/s the round-trip window is 11 KB, whereas at a speed of 1.7 Gb/s the round-trip window is 12 MB.

We propose procedures whereby a high-speed wide-area virtual-circuit network dynamically adjusts cell window sizes as the traffic changes, so that each user has a window large enough to support a fair share of network bandwidth, but not necessarily a full round-trip window. We maintain network stability under congestion by providing signaling protocols whereby users who follow the protocols will never lose data because of buffer overflow within the network. A related scheme has been analyzed by Doshi and Heffes \cite{12}.

2. NETWORK ARCHITECTURE

In the network of Fig. 1, each switching node has a memory buffer. The buffer is logically divided into a per-circuit buffer for each virtual circuit, together with some unassigned memory space. The per-circuit queues are served round robin. The amount of buffer space assigned to each virtual circuit may vary from time to time but is never less than the circuit's current window size. The switching controller maintains a list of the amount of memory allocated to each virtual circuit. If a virtual circuit's buffer overflows, data belonging to that virtual circuit only is discarded.

The routers enforce edge-to-edge cell windows on the virtual-circuit network. Each virtual circuit is given a default window when it is set up, and it retains a window of at least this size for as long as the circuit exists. The input router can issue requests for increases in window size up to a full round-trip window, or decreases down to the default window. Increases may be requested at intervals as short as one round-trip time; decreases are requested at longer intervals determined by a renegotiation timer. A request travels around the virtual circuit and the switch controllers confirm or modify it, at the same time making appropriate changes in buffer allocation. When the response returns, the input router knows that allocations corresponding to a new window exist all around the circuit. The router begins to use a larger window after the increased buffer allocations are confirmed. It begins to enforce a smaller window before requesting a decrease in buffer allocations.

3. WINDOW SIZING

A single user on an otherwise idle system needs a window of size $W_0$ in order to use the full network bandwidth. A user who shares the network with other active users does not need a window as large as $W_0$. In principle, if $N$ users transmit continuously, each of them needs a window no larger than $W_0/N$, and the total window requirement is just $W_0$.

As a slightly more realistic example, consider the hypothetical case in which circuits are turned on at a time and transmit continuously. If we assign a window of size $W_0/k$ to the $k$th circuit,
then when \( k \) circuits are transmitting, the round robin queueing discipline gives each one a fraction \( 1/k \) of the trunk bandwidth, and nobody's throughput is unnecessarily window-limited. If the \( k \)th user gets a window of size \( W_0/k \), the total buffer space \( B \) for \( N \) users satisfies

\[
B / W_0 = \sum_{k=1}^{N} 1/k \approx \log N. \tag{1}
\]

In general, users of data networks may be classified into three broad categories: 1) Interactive users, whose transmissions are shorter than the default window; 2) bulk users, whose transmissions are long compared to \( W_0 \) and are separated by intervals long compared to \( T_0 \); and 3) intermittent users, who exhibit periods of activity marked by repeated transmissions comparable to \( W_0 \) at intervals comparable to \( T_0 \), separated by periods of inactivity long compared to \( T_0 \). Intermittent users are the hardest to provide for efficiently, and it is this class of users that we consider in sizing windows.

The model that led to Eq. (1) is unrealistic in that it assumes that the number of users who are actively transmitting when a new user arrives is equal to the number of users who are holding window allocations as large as or larger than the new user. The average user does not transmit during some part of the time that it holds a large window. We model the effect of idle time as follows.

Construct a closed queueing network model with \( N \) users and three nodes or states: A (actively transmitting), I (idle but holding a large window), and Q (quiescent, without a large window). Users circulate between nodes A and I, with occasional excursions from I to Q to A. Node A is single-server with processor sharing, while I and Q are infinite-server nodes. The mean service requirements per visit to nodes A and I are \( \tau \) and \( h \tau \), respectively. We would like to determine the expected number \( J_k \) of active users, given that \( k \) users are currently holding large windows, i.e., are in either state A or state I. The conditional distribution of these \( k \) users matches that of a closed network containing only nodes A and I and \( k \) users. Mean-value analysis [13] of this two-node loop shows that a user's average activity time is \( (J_k - 1 + 1) \tau \) out of an average cycle time of \( (J_k - 1 + 1) \tau + h \tau \); hence

\[
\frac{J_k - 1 + 1}{J_k - 1 + 1 + h} = \frac{J_k}{k}. \tag{2}
\]

It is trivial to evaluate \( J_{k-1} \) numerically from the recurrence (2) for any value of \( h \), starting from \( J_0 = 0 \). However, a remarkably accurate closed-form expression over the parameter ranges of interest can be obtained by replacing the right side of (2) by \( J_{k-1} / (k - 1) \) and solving the resulting quadratic equation for \( J_{k-1} \). This analysis can be used to size windows as follows. Suppose that the \( k \)th arriving user receives a window allocation \( W_{k-1} \), while the \( k - 1 \) users already present have windows greater than or equal to \( W_{k-1} \). The convention of denoting the \( k \)th user's window by \( W_{k-1} \) permits us to denote the first user's window by the symbol \( W_0 \) that we have used for the full round-trip window. In order that the \( k \)th user's throughput not be unnecessarily restricted, \( W_{k-1} \) needs to satisfy

\[
W_{k-1} / W_0 = 1 / (J_{k-1} + 1). \tag{3}
\]

Using the closed-form approximation to \( J_{k-1} \) yields

\[
\frac{W_{k-1}}{W_0} = \frac{2}{k-h + [(k-h)^2 + 4h]^{1/2}}. \tag{4}
\]

If the \( k \)th user is given an allocation according to (4), the total allocation for \( N \) users satisfies

\[
\frac{B}{W_0} = \sum_{k=1}^{N} \frac{2}{k-h + [(k-h)^2 + 4h]^{1/2}}. \tag{5}
\]

Approximating the sum by an integral gives

\[
\frac{B}{W_0} = \int \frac{x_2}{r_2 + x_2} - \frac{x_1}{r_1 + x_1} + \log \left[ \frac{r_2 + x_2}{r_1 + x_1} \right], \tag{6}
\]

where \( r_{1,2} = (x_{1,2}^2 + 4h)^{1/2}, \ x_1 = 1/2 - h, \) and \( x_2 = x_1 + N \). A little algebra shows that

\[
\frac{B}{W_0} = \log N + h/2 \tag{7}
\]

for \( N \gg h \gg 1 \).

What are realistic values of \( h \)? Presumably the intermittent users will have message lengths no shorter than about one round-trip window, because if they had much shorter messages they would not have been allocated maximal windows. The renegotiation timer enforces a cutoff on the length of idle intervals. The cutoff could be as short as about one round-trip time; however the network will probably allow users a few seconds of think time before reducing their windows. A transcontinental round trip is about 60 ms. An average idle time of 6 s per round-trip window of transmitted data corresponds to \( h = 100 \).

Figure 2 shows total buffer usage computed from mean value analysis (MVA) and from the approximation (4). The difference is everywhere less than 4%. The fact that the curves have a knee near \( N = h \) is plausible, because if the average circuit is idle for \( h \) times as long as it is active, about \( h \) circuits can coexist before circuits begin seriously
competing with each other. Total buffer usage increases slowly beyond \( k = h \).

4. SIGNALING PROTOCOLS

Requests for changes in cell window size on a connection from host \( H_A \) to host \( H_B \) in Fig. 1 are initiated by the input router \( R_A \). If there are only a small number of window sizes, signaling information can be written into a congestion control field in the header of each ATM cell. One bit is a request-response bit, one bit is a sequence number, and the remaining bits code the window size. A router continues to repeat a request bearing a given sequence number until it receives a response bearing that same sequence number. If there are too many window sizes to be coded into the available header space, congestion control information may be carried as data in special congestion control cells.

For simplicity, assume that the input router can choose its request from a continuum of sizes, that is, any number of cells ranging from the default window up to a full round-trip window. Similarly, a switch controller has a continuum of possible responses.

The transmitting router is allowed to request an increase in window size at any time after the response to the previous request has been received. To determine the size of the next window request, the router keeps track of the total number of cells that the given virtual circuit has in the system, that is, the contents of the input packet queue plus the unacknowledged data in the cell window. When the router is able to request an increase in window size, it requests a window, up to the full round-trip window, large enough to hold all of the foregoing data, provided the amount exceeds the current window.

The transmitting router also keeps track of the cumulative throughput of the virtual circuit during the current renegotiation interval. When the renegotiation timer expires, the router requests a window, down to the default window, that is just large enough to contain the data that the virtual circuit has instantaneously in the system, and also to accommodate the average throughput that the circuit has experienced during the previous renegotiation interval. The last provision prevents the closing down of a circuit's window if the circuit has momentarily gone idle after recent activity.

If the transmitting router cannot accommodate a full round-trip window of data in the input packet buffer of each virtual circuit that it supports, the router should provide an input buffer at least as large as the current cell window for the virtual circuit, and it should keep track of input buffer overflows. An input overflow since the last request for an increase in cell-window size can be treated as if the circuit had a full round-trip's worth of data in the system at the time the next request for an increase in window size goes out.

Each switch controller has a table of the function \( W_{k-1} \) defined by (4). The controller keeps track of the buffer allocations of its virtual circuits, and also the rank of each virtual circuit, where the rank is the ordinal number of that circuit's buffer allocation. The circuit with the largest allocation has rank 1, and so on. The current allocation of the circuit whose rank is \( k \) does not exceed \( W_{k-1} \).

When a request arrives to change the buffer allocation of a given virtual circuit, the controller goes through an “off-line” computation, and the results are written into cells that pass through after the computation is complete. Suppose that there is a request to increase the buffer allocation of a given virtual circuit to \( W_{r'} \). The circuit is assigned a hypothetical allocation \( W_{r'} \), and its rank is exchanged with circuits having higher ranks until consistency is achieved or until further exchanges are impossible because they would require demoting some other circuit to a rank inconsistent with that circuit's current allocation. Let \( k \) be the final rank of the given circuit. The circuit receives an allocation equal to \( \min(W_{r'}, W_{k-1}) \), and the final ranks are consistent with the final allocations. If a decreased allocation is requested, the decrease is given automatically and the circuit's rank is exchanged with circuits having lower ranks until consistency is achieved. The new window size is then passed to the next switch.

The receiving router \( R_B \) observes the consensus window allocation of the switch controllers along the forward path, changes the request-response bit to "response", and copies the consensus allocation into cells returning to the input router.

On the return trip, each switch controller sees the consensus allocation. If the consensus is less than the allocation that the given controller is already holding for the virtual circuit, the controller reduces its allocation and adjusts ranks
accordingly. When a cell bearing the consensus allocation returns to the transmitting router, the router begins to use the new window if it is not already doing so. The router is then free to initiate a new request bearing the other sequence number.

5. PERFORMANCE ISSUES

Network instability due to cell loss within the virtual-circuit network should not occur, since the buffer-management scheme is designed never to lose cells. Losses may occur, however, at the packet input queues. It is important to understand how host-to-host retransmission protocols interact with the congestion management algorithm on the virtual-circuit network ([15]).

Many TCP/IP hosts use the Jacobson-Karels (JK) retransmission protocol [4]. This protocol starts with a window size of one packet and increases the packet window size each time an acknowledgment is received up to a maximum size called the receiver window, which is generally unrelated to the round-trip window of the cell network. The packet window size is decreased to one packet whenever a retransmission is required. The timeout interval is doubled, or at least not decreased, after each timeout.

On a new connection, JK opens the packet window at a rate compatible with the opening of the cell window by the virtual-circuit network. However, JK may eventually open the packet window wider than a cell round-trip window, and in any case it does not decrease the packet window until there is a retransmission timeout. A virtual circuit’s input buffer, on the other hand, is unlikely to exceed one round-trip window, and the cell network may reduce the size of the input buffer if the circuit becomes quiescent. If the JK packet window exceeds the network cell window, JK may drop a series of packets and take a long time to recover.

A simulator has been written for networks of hosts, routers, and switches, in which hosts implement the JK algorithm. The size of the input packet buffer for each connection is chosen according to one of two assumptions: 1) a dedicated packet buffer of fixed size equal usually to one round-trip cell window, or 2) a dynamic packet buffer equal to the current cell-window size on the given virtual circuit. Incoming packets that would overflow the input buffer are dropped.

i) Effect of window-size parameter. Figure 3 assumes 10 connections sharing a single trunk link, with infinite input buffers and dynamic cell-window management but no host-to-host flow control. Each source generates messages with a mean of approximately one round-trip time, and think times adjusted so that the total offered load is 100%. The renegotiation interval is 50 round-trip times, with windows sized according to various values of the window-size parameter \( h \). The figure shows the increase in mean trunk utilization as a function of \( h \). The theoretical utilization of a product-form network with 10 users sharing a single processor-sharing node, no flow control, and an offered load of 100% is indicated by the \( x \) at the edge of the plot. The knee in the curve near \( h = 9 \) is reasonable, since each user’s average idle period is 9 times its average service requirement.

ii) Interaction between cell windows and packet windows. Figure 4 relates to a single user on an otherwise idle trunk. The user generates messages with a mean of one round-trip time, and think times adjusted to give an offered load of 40%. The messages are flow-controlled both by JK and by the dynamic cell-window management algorithm. JK has a receiver window approximately 10 times as large as the network round-trip cell window. The size of the router’s input packet buffer is adjusted so that it is always equal to the current cell window. The top panel of Fig. 4 shows typical behaviors of the packet and cell windows when the renegotiation interval is 50 round-trip times. JK has no advance notice of reductions in the input buffer size and must retransmit all the dropped packets. When JK encounters a series of dropped packets, a situation for which the original algorithm was not designed, it goes through a period during which one dropped packet is retransmitted at a time and the interval between retransmissions continually doubles. Only after the last dropped packet has been retransmitted does the protocol return to its normal slow-start procedure.

The lower panel of Fig. 4 shows the effect of adding “source quench” to the router protocol. In this mode, when a packet is dropped at the input buffer a NACK carrying the sequence number of the dropped packet is immediately returned to the source host. The host reinitiates the slow-start procedure, so that typically only one or two packets have to be retransmitted; long pauses do not occur.

We report elsewhere [14] the results of further studies showing the effects of the window-size parameter, and also the interactions between

![Fig. 3. Effect of window-size parameter.](image-url)
cell windows and packet windows for multiple users on a single link and on tandem links. The results confirm expectations. Packet flow control and cell flow control can interact destructively unless there is some communication between routers and hosts.

6. CONCLUSIONS

We have described an approach to dynamic window sizing on a virtual-circuit network with limited buffer space, which takes into account the fact that users do not transmit during some part of the time that they hold a window allocation. We have also described a simple scheme for deciding when a request for increase or decrease of window size should be made; the decision process does not require collaboration by users of the network. We have designed a signaling procedure that maintains a consistent view of buffer allocations across the network, so that data cells are not lost due to buffer overflow on any virtual circuit. Limited numbers of simulations have shown that the network protocols function as expected. However, if the end users of the network employ independent packet retransmission strategies, the end-to-end performance of the system may be poor, because under congestion the entrance routers drop user data at the edge of the virtual-circuit network. One way to restore fair, stable, and efficient performance is to send a source-quench signal from entrance router to host when a packet is dropped.

It is a pleasure to thank Bharat Doshi for discussions of many issues related to this work.

REFERENCES