MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Buffering in an ATM Local Area Network With Real Time Channels

Hugh C. Lauer

TR93-13 December 1993

Abstract

Traffic in local area networks is characterized by considerable b̈urstiness, even traffic representing continuous data types such as audio or video which require real-time network guarantees. Existing algorithms for implementing real-time channels assume large or unlimited buffer space in network switches in order to buffer entire messages. However, such large buffers also imply large worst-case delays and high jitter. An alternative approach in an ATM network is to move the buffering to the client interface and to transmit cells of a real-time channel more uniformly. This allows the network to guarantee tighter deadlines and less jitter and to reduce the amount of reserved buffer space required to support the real-time channels at each switch. It is argued that this is practical in local area networks, even when it is not always practical in wider area networks.

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 1993
201 Broadway, Cambridge, Massachusetts 02139

Mitsubishi Electric Research Laboratories
Cambridge Research Center

TR93-13 July 16, 1993

Buffering in an ATM Local Area Network With Real Time Channels

Hugh C. Lauer

Abstract

Traffic in local area networks is characterized by considerable "burstiness", even traffic representing continuous data types such as audio or video which require real-time network guarantees. Existing algorithms for implementing real-time channels assume large or unlimited buffer space in network switches in order to buffer entire messages. However, such large buffers also imply large worst-case delays and high jitter. An alternative approach in an ATM network is to move the buffering to the client interface and to transmit cells of a real-time channel more uniformly. This allows the network to guarantee tighter deadlines and less jitter and to reduce the amount of reserved buffer space required to support the real-time channels at each switch. It is argued that this is practical in local area networks, even when it is not always practical in wider area networks.

Extended abstract submitted to
Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, November 1993.

Mitsubishi Electric Research Laboratories, Inc.,
201 Broadway, Cambridge, Massachusetts 02139

Revision History:–

1. First version, TR 93-13, *July 16, 1993*

Copyright © Mitsubishi Electric Research Laboratories, Inc., 1993
201 Broadway, Cambridge, Massachusetts 02139

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of the Mitsubishi Electric Research Laboratories of Cambridge, Massachusetts; an acknowledgment of the authors and individual contributors to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to the Mitsubishi Electric Research Laboratories. All rights reserved.

1 Introduction

Let us consider a fast local area network with very fine-grain switching — e.g., a network based on Asynchronous Transfer Mode (ATM) technology. We want to support traditional kinds of computer network traffic, including

- request-response communication, typically characterized by small messages with very low latencies, and
- bulk data transfer, in which large amounts of data must be transferred from one place to another, on demand, as quickly as possible.

In addition, we wish to support a third class of communication, namely continuous media such as digitized audio or video.

Much has been written about establishing *real-time channels* within a network in order to provide guaranteed performance, bandwidth, delay, and/or jitter characteristics to clients with real-time or continuous data requirements [Ferrari90, Zheng93]. In a typical situation, a client makes a request to establish a real-time channel with parameters

$$(M, T, D)$$

where M indicates the maximum size of a message (in ATM cells), T indicates the minimum interval from the start of one message to the start of the next message, and D indicates the maximum acceptable end-to-end delay of the message. The network then determines whether or not it can guarantee the performance of the channel, based on deadline scheduling of cells, bandwidths of links, buffer capacities of nodes, and outstanding guarantees to previously established real-time channels.

An open question is how much buffering to provide in the network nodes. It is generally accepted wisdom that local area network traffic — including real-time traffic — is very "bursty." For reasons explained below, this leads to the requirement of large buffers in the nodes or switches of the network. It also leads to large worse-case delays at each switch and to potentially large jitter in the arrival of real-time data at its destination.

An alternative is to move the onus for buffering back to the client, or at least to somewhere in the client's protocol stack, operating system, or network interface. In effect, this would smooth out traffic on the real-time channel by converting it to traffic with parameters something like

$$(M/k, T/k, D')$$

for some value of $k > 1$. This paper explores some consequences of this alternative.

2 Worst Case Analysis

Consider a single switching node in the network, and suppose that n real-time channels coming into the node are all routed to the same output link. Suppose also that each input

link j can transmit data at the rate of R_j cells/second and that the output link can transmit data at the rate of R_0 cells/second. In the worst case, a message arrives on each real-time channel at the same instant. Cells start to fill up buffers in the switch at the rate of

$$\sum_{j=1}^{n} R_j = R_{input} \text{ cells/second}$$

After a small, constant delay κ to dispatch the first cell, cells start to drain out of the buffers at the rate of

$$R_0 \text{ cells/second}$$

If $R_{input} < R_0$, then cells drain away faster than they accumulate. Only a constant amount of buffer space is required within the switch and only a constant delay is introduced by the switch. But if $R_{input} \geq R_0$, then the buffers will fill up at the rate of $R_{input} - R_0$ cells/second for at least as long as it takes to receive one complete message — i.e.,

$$\tau = \min \{ \frac{M_j}{R_j} \} \text{ seconds}$$

This suggests a lower bound on the amount of buffer space which must be reserved in the switch in order to guarantee that none of the n real-time channels suffers cell loss, namely

$$\tau \times (R_{input} - R_0) \text{ cells.}$$

In actual practice, it is probably better to provide enough buffer space in the switch for one full message on each channel — i.e., $\sum M_j$ cells. If, for example, messages are individual frames of a video stream, this can represent a lot of cells.

Large buffers of sizes imply long delays in the worst case and, consequently, high jitter in the normal case. In the worst case (assuming that the channels are numbered in deadline order), the last cell of channel n leaves the switch at least $(\sum M_j) / R_0 + \kappa$ seconds after the first cell arrives (more if other channels have very small T_j). Thus, the best deadline D_n that the switch could possibly guarantee is

$$D_n = (\sum M_j) / R_0 + \kappa.$$

On the other hand, if a message on channel n arrives when there are no other messages contending for the same output link, then the last cell leaves the switch $M_n / R_n + \kappa$ seconds after the first cell arrives. Thus the switch introduces a maximum jitter of at least

$$(\sum M_j) / R_0 - M_n / R_n$$

into channel *n*. Depending upon the kind of traffic on the channel, this will demand a corresponding amount of buffering at the destination to restore a continuous data stream.

3 Impact on the Network

Note that providing enough buffer space is not really an issue. Memory for buffers is cheap compared to the cost of the rest of the components of an ATM switch, and is likely to remain so for a long time. However, delay and jitter are not "cheap," but rather cumbersome and onerous. In the previous section, it is apparent that the best delay and jitter that an individual switch can offer grow proportionally with both the number of real-time channels contending for each output link and also with the maximum size of a message or burst.

Note also that both the delay and jitter are cumulative. That is, the delay and jitter of a real-time channel passing through a number of switching nodes grows as it traverses the network, contending with other real-time channels over intermediate links. Furthermore, jitter in one switch increases the buffer requirements in subsequent switches of a path through the network, as shown in [Zheng93].

Consider an example:— suppose all *n* real-time channels support video data with frames of up to one megabit (i.e., about 2600 cells) 30 times per second, and suppose that all of the links operate at 155 megabits/second (about 366,000 cells/second). Let κ be small, say, about two cell times or 5.5 microseconds. Multiplexing three such channels onto one output link requires about 65% of bandwidth of that output link (in the worst case). Thus, the switch will need up to 7800 cells of buffer space for the three channels. The worst case delay is about 22 milliseconds (about two-thirds of a frame time) and the jitter is about 14 milliseconds. This is the delay and jitter introduced by a single switch. If one channel passes through, say, three hops and is multiplexed with two new real-time channels of the same characteristics at each hop, the cumulative delay would be nearly two frame times and the jitter would be almost one and one-half frame times.

By contrast, suppose we are able to smooth out the traffic on real-time channels by buffering in at the interface or protocol stack. Then a client's original request for a channel with parameters (*M, T, D*) would be transformed by the system into a network request for a channel with parameters (M/k, T/k, D').

In the extreme, let $k = M$ — i.e., suppose that the maximum-sized message is transmitted one cell at a time, uniformly spaced over the minimum interval *T*. Then the maximum buffer requirement to support *n* real-time channels over a single link is *n* cells (plus the constant amount to cover the cell switching delay). The best delay of each cell that the switch can guarantee to the client of the *n*th channel is

$$D'_n = n/R_0 + \kappa,$$

where we again assume that channels are numbered in deadline order. The maximum jitter for channel *n* is $n/R_0 - 1/R_n$. The last cell of a client message originally of size M_n

would then leave the switch approximately $T_n + D'_n$ seconds after the first cell of that message arrives.

Let us revisit the example of three video channels being multiplexed over one output link. The switch requires only three cells of buffer space, plus enough to support the constant switching delay κ. Cells from each channel arrive at a rate not exceeding one every $1/(30 \times 2600)$ seconds — i.e., every 13 microseconds — and leave within about $8 + \kappa = 13.5$ microseconds. It takes a full frame time to transmit each frame through the switch, but the maximum jitter is only about 5.5 microseconds. Furthermore, since full frames are not being buffered in each switch, the cumulative delay and jitter through a network is limited to the cumulative delay and jitter of a cell, not a whole frame. In the case of three hops, a video frame still takes a full frame time to traverse the network, but experiences only a maximum of $3 \times (8 + \kappa) = 40.5$ microseconds of delay and about 16.5 microseconds of jitter.

4 Notes and Comments

One example is by no means sufficient for defining a network protocol and policy. Nevertheless, it appears that by moving the buffering of real-time channels out of the network and into the interface, better service can be offered. Philosophically, we can look at it from a system point of view. In order for the network to be able to handle bursty traffic from multiple clients and still provide reasonable guarantees, buffering is needed *somewhere*. If the network itself takes on the responsibility to provide the buffering for the largest real-time message, it also limits itself in its ability to guarantee tight delays and low jitter. If, however, the responsibility is moved back to the client interface, smoothing out the bursts at that point, then the delays and jitter introduced by the network are much less.

Is it reasonable to build a local area network on these assumptions? I believe that it is. In practical networks, clients do not interface directly with the network but with a protocol stack, typically involving an operating system and a network interface device with some degree of intelligence. Among other things, these layers partition high-level messages into cells and perform other overhead and housekeeping functions. Except in the rarest cases, the original information is already buffered, either in the client application itself or somewhere else in the stack. The computational cost of transmitting real-time channel cells at a measured rate is negligible and trivial to implement.

An analysis of the impact of this policy on other kinds of local area network traffic is beyond the scope of this short paper. Intuitively, however, it seems favorable, especially to bulk data transfer. In general, when a client requests a bulk data transfer connection, it is asking for (its fair share of) the maximum available network bandwidth. The implementation of such connections necessarily requires sophisticated flow control, so that the data rate is both

- not so great that it creates congestion within the network, thereby causing information to be discarded and hence requiring retransmission, and

- not so small that it leaves network bandwidth underutilized, thereby causing its own transfers to take longer than necessary.

On-demand bulk data traffic is, by definition, of lower priority than the real-time channels *guaranteed* by the network. However, it seems that it would be much easier and more practical for the flow control algorithms to dynamically set an optimal bulk data rate if the burstiness of the real-time channels were less and the traffic were more predictable.

Note that this analysis was specifically oriented toward local area ATM networks and does not apply to long haul networks. First, the small constant switching delay per cell (a few microseconds, which we afford to ignore in the local case) is dominated by transmission delay (tens milliseconds for coast-to-coast traffic) and framing delays typical of long-haul telecommunications networks. Second, long haul networks are usually optimized for bandwidth, not latency — indeed, telecommunications companies make their money by selling bandwidth. Traffic patterns, resource allocation, etc., are characterized by the Law of Large Numbers. By contrast, local area networks are characterized by the Law of Small Numbers and thus must approach the same problems quite differently.

The dream of a seamless ATM network spanning the department, campus, metropolis, and country is, I believe, a pipe dream. There will be not only different administrative and financial pressures but also different technical pressures at the different levels. These will affect how resources are allocated, what kinds of traffic will be accepted, what kinds of guarantees can be offered, etc. It is almost inevitable that there will be gateways between administrative boundaries. In practical networks, these gateways will not only do protocol processing and routing, but also message buffering, splitting and reassembly, and other management to adapt to the characteristics of the networks on each side.

Acknowledgments

The philosophy in this paper grew out of many discussions with John Howard, Randy Osborne, and especially Chia Shen and Qin Zheng, who each spent many hours explaining to me what real-time channels really are.

References

[Ferrari90]
D. Ferrari and D. Verma, “A Scheme for Real-Time Channel Establishment in Wide-Area Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 8, n. 3, April 1990, pp. 368-379.

[Zheng93]
Q. Zheng, K. G. Shin, and C. Shen, “Real-time Communication in ATM Networks,” submitted to *Real-Time System Symposium*, 1993.