Impact of ATM switch architectures on CBR video performance

S. Zeadally *

Department of Computer Science, Wayne State University, Detroit, MI 48034, USA

Received 5 July 1999; accepted 5 July 1999

Abstract

Asynchronous Transfer Mode (ATM) has become one of the leading high-speed networking technologies in the local and wide area environments. ATM offers great promise to meet the real-time requirements of emerging multimedia applications. In the last few years, many ATM networks have been deployed and use different types of ATM switches, many of which differ in their implementations. In this paper, we explore the impact of two ATM switches, namely DEC GIGAswitch/ATM and IBM 8265 on the performance delivery of real-time digital video over native ATM connections. We also demonstrate experimentally the effects of different types of cross-traffic (delivered over both UDP/IP and native ATM) network loads on end-user application performance using QoS metrics such as jitter and packet loss. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Asynchronous transfer mode; Jitter; Performance; Switch; Video

1. Introduction

Asynchronous Transfer Mode (ATM) is a connection-oriented switching and multiplexing technique that uses small 53-byte packets, called cells, to transfer different types of data. It is the first communications technology to carry audio, video and other data in a unified way on one common platform [1,2]. ATM supports Quality of Service (QoS) and different service classes namely Constant Bit Rate (CBR), Variable Bit Rate (VBR), Unspecified Bit Rate (UBR) and Available Bit Rate (ABR) [3]. All these features of ATM along with its high bandwidth, low latency, and versatility are appealing for network-based multimedia applications. This has resulted in ATM becoming a popular, high-speed network choice both in the local area and wide area environments in recent years.

Real-time delivery of continuous media such as audio and video needs strict timing requirements to be met by the underlying network and end-system to ensure smooth delivery of these media to the end-user. Therefore, QoS parameters such as jitter (the variation in the delays experienced by packets transmitted over the network) and cell loss must be kept within bounded limits to ensure that data is delivered continuously. Otherwise, if jitter is not kept within tolerable limits, we experience video flickering, audio distortions, and possibly temporary blank screens in the case of images. Previous works on packet switched networks have been based on either analytical methods or simulation models to predict performance. Computation of parameters such as jitter analytically [4–8] is extremely complex as it involves the determination of the characteristics of the output process [9] departing from the queuing system. Further, simulation models cannot capture all features of an actual network system. In a real networking system, there are many hardware and software components whose interactions are very difficult to predict by fixed models or simulations.

In this work we choose real-time digital video as our continuous medium because it is interesting due to its high bandwidth demand, real-time delivery, and intensive processing requirements. The goals of this work are to investigate the following issues using real-time CBR video traffic:

- The effects of ATM switch architectures, in particular the buffering and flow control strategies on end-to-end user application jitter and packet loss performance.
- The effects of cross-network traffic loading on end-user application jitter and packet loss for CBR video traffic.
- The effects on end-user jitter performance when transmitting CBR video over native ATM compared to UDP/IP over ATM.
- The impact of end-to-end network bandwidth reservation (using native ATM) on video jitter and packet loss in loaded network conditions.

The remainder of this paper is organized as follows. Section
2 presents background information and related work. In Section 3, we describe the experimental testbed and measurement procedures. Section 4 discusses our experimental results. Finally, in Section 5, we make some concluding remarks.

2. Background and related work

Owing to the statistical nature of packet traffic, in many packet switches, packets/cells are likely to contend for the same output port. Queuing is therefore important in these circumstances to provide contention resolution. The queuing architecture of an ATM switch is of fundamental importance to the capabilities of the switch. Queues can be placed either before (referred to as input queuing) or after the routing functions (referred to as output queuing). From a performance viewpoint, output queuing has been shown to achieve better throughput than input queuing [10–12], but with actual implementations, FIFO queues with multiple ports are hard to implement on a single VLSI chip. They are expensive compared to single-port FIFO buffers. Relying solely on output queues also has the disadvantage that in the presence of bursty traffic, the depth of the output queues depends on the number of ports sending cells to the queue. Concurrent bursts are likely to increase with the number of ports which implies that switches relying solely on output queues may not maintain consistent performance levels with increase in the number of ports unless additional buffers are provided which lead to increased cost and complexity. On the contrary, switches built with input queues only are less efficient because no output traffic shaping capabilities are possible for physical or logical ports (i.e. Virtual Path) as the traffic for a given output port is actually distributed in various input ports.

Various ATM switch architectures have been proposed in recent years [13–15]. However, in this work we used only two switches namely, the DEC GIGAswitch/ATM and the IBM 8265. We describe below the main features of these ATM switches.

- **DEC GIGAswitch/ATM switch**: The DEC GIGAswitch/ATM [16] (Fig. 1 (left)) is a high-performance switch which is based on the design of the AN2 switch [17]-a 16 × 16 switch fabric. The DEC switch consists of a 16-port synchronous, non-blocking crossbar and can support up to 16 line cards. Each port can hold either a 4-port OC3 (155.52 Mbits/s) line card or a one-line OC-12 (622.02 Mbits/s) card. The bandwidth of the switch is 800 Mbits/s per port (full-duplex) which results in an aggregate throughput of 12.8 Gbits/s. However, in our experiments we used a DEC GIGAswitch/ATM system with 14 slots, one of which is taken by the control module, leaving a total of 13 slots available for line cards (i.e. an aggregate throughput of 10.4 Gbits/s). The buffer size on each line card is about 0.5 Mbytes and provides buffering for approximately 10,240 ATM cells. The DEC switch uses per virtual circuit random access buffering in the input units in order to avoid head-of-line blocking normally experienced in FIFO-buffered switches. Quota counters limit the amount of buffering that can be consumed from the main buffer. In our DEC switch, the quota used per OC-3 port is 2000 ATM cells. The quota counters are provided so that switches or hosts cannot unfairly exhaust the entire buffer pool shared with other (Virtual Circuit) VCs. DIGITAL’s GIGAswitch/ATM system uses advanced queuing management to ensure non-blocked switching. The SWITCHmaster [16] function uses DIGITAL’s patented distributed algorithm called Parallel Iterative Matching Technique that involves both input and output units in order to arbitrate access to the crossbar switch. A credit-based flow control scheme is used to pass data cells from the input units to the output units. The output units have a limited buffering for each line (around 157 cells). With the DEC switch, cell loss never occurs at the output units because before cells from the input are allowed to be transferred through the crossbar, the output units must send credits to the input unit (Fig. 1). In other words, the output units communicate flow-control information with the input units and regulate the cell flow through the crossbar switch fabric.

Different scheduling implementations are used for handling ATM traffic transported over UBR VCs and CBR VCs in the DEC ATM switch. Valid cells arriving on
UBR Switched Virtual Circuits (SVCs) are placed at the end of the VC’s cell queue. If the VC is active¹ (with a non-zero credit balance where each credit corresponds to an available buffer), it is placed on the port queue corresponding to the required output port. UBR traffic complete for the switch capacity not used by CBR traffic, using a distributed arbitration algorithm that ensures high utilization of the switching fabric. However, cells transmitted over CBR VCs are not placed on port queues but are transmitted according to a fixed schedule. CBR VC cells flow through the switch under control of a Time Division Multiplexing schedule. There are two CBR schedule tables. At any given time, only one of the tables is used for scheduling. The other is filled with a new schedule when a new CBR VC is added to or removed from the schedule. When the new schedule is ready, the master line-control processor of the switch signals all line cards to start using a new schedule.

• **IBM 8265 ATM switch**: The IBM 8265 ATM switch [18,19] (Fig. 1 (right)) uses IBM’s switch fabric which delivers an aggregate throughput of 12.8 Gbits/s on a 25 Gbits/s ATM backbone. The 8265 switch design includes single-stage switching, distributed buffer pools, traffic management functions such as traffic shaping at the VP level. The IBM 8265 ATM switch combines the strength of central switching fabric design with sophisticated distributed buffer pools and traffic management. The architecture is based on a single-stage, 16 × 16 switching fabric. A single-stage switching design is a better solution than multi-stage designs that introduce higher jitter and cell delays. The switching fabric consists of two IBM PRIZMA ASICs [20] which deliver 800 Mbits/s (full-duplex) per port with an aggregate throughput of 12.8 Gbits/s on a 25.6 Gbits/s ATM backbone that is star-wired. The PRIZMA switch fabric (called ‘Switch-on-a-Chip’) switches fixed size cells and is equipped with output buffers to manage contention on the output ports (Fig. 1). The IBM 8265 ATM switch combines both input and output queuing in its implementation. The IBM switch uses a combination of output and input queuing by providing moderate amounts of expensive memory at the output units with larger amounts of less expensive memory at the input units. The output memory is shared with all output queues. To ensure fairness and avoid the entire shared memory being consumed by one output queue (thereby degrading performance of connections to other output queues), the maximum amount of shared memory allocated to an output queue is restricted.

The IBM switch exploits a leaky bucket function to maintain transmission efficiency for CBR traffic. The policing function used smoothes the input traffic on a per connection basis and ensures that the CBR SVC connection’s peak cell rate agreed is not exceeded.

It is also worthwhile noting that both IBM and DEC ATM switch architectures support early packet discard and partial-packet discard schemes.

Related work, which investigated QoS parameters for multimedia networks, had been undertaken by the Tenet Group (University of California at Berkeley). Schemes for jitter control in packet-switched networks are discussed in [21–23]. Other schemes for guaranteeing jitter bounds for high-speed networks in the network and application layers have been studied by Kadur et al. [24].

3. Experimental network testbed environment

The experimental network testbed for our local area ATM network consists of four Intel Pentium II 400 MHz Personal Computers (PCs) which are connected to two ATM switches using the configuration shown in Fig. 2. It is worthwhile noting that as we only have one DEC ATM switch, it was not possible to experiment with a network configuration of two DEC switches connected to each other. A pair of PCs was used to send and receive live, uncompressed video. Another pair of PCs was used to load the switch. Our goal was to investigate the impact of ATM switch architectures with different network loads (causing output port contention on the link between the ATM switches) on video jitter and packet loss performance at the application level. The physical

---

¹ VCs may be enabled or disabled by the line control processor of the switch. Disabled VCs are never placed on a port queue for output.
link connecting the ATM switches and PCs uses multimode fiber running at 155.52 Mbits/s. The ATM device drivers used in our PCs are from Efficient Networks Inc. and also support OC-3 rate ATM connections. The Efficient ATM device driver supports both Classical IP [25] and native ATM (over AAL5). All the PCs used in our tests run Windows NT 4.0 operating system, and equipped with 256 Mbytes of Random Access Memory (RAM), and one 4 Gbyte disk drive. It is worth noting that the Windows NT networking Application Programming Interface (API) namely Winsock 2 [26] provides full native ATM support. As a result, all our native ATM applications are based on standard Winsock 2 API that make them portable. Further, as Winsock 2 API supports native ATM, we are able to implement QoS support in our applications. Currently, the ATM device driver supports only bandwidth reservation with peak rate allocation for CBR traffic. Both the DEC and IBM ATM switches used in our experiments support dynamic bandwidth reservation at connection set-up time using User Network Interface (UNI) 3.1 and ATM signaling. Thus, for CBR native ATM connections, we were able to reserve network bandwidth on an end-to-end basis via the ATM switches. All connections (CBR and UBR) were made using SVCs.

We implemented a video conferencing system that delivers live, uncompressed video at different resolutions over both UDP/IP and native ATM via the Winsock 2 API. The basic operations used for sending live video over the network are: live video is captured by a video frame grabber and transferred in its uncompressed form to host kernel memory by the video capture board. The video conferencing application copies the video data from the kernel buffer to a socket buffer in user space. To deliver the video over the ATM network, it is necessary to copy the data back to a kernel network buffer. Finally, the ATM device driver transmits the video data over the ATM network. A more detailed description of the system is given in [27,28]. Our video conferencing application can deliver video (16-bit pixel) at different frame rates and resolutions: at 25 frames/s (throughput of 7.7 Mbits/s) with 160×120 pixels; at 25 frames/s (throughput of 30.7 Mbits/s) with 320×240 pixels, and at 19 frames/s (throughput of 93.4 Mbits/s) with 640×480 pixels. In all experiments described below, we transmitted 640×480, 16-bit pixel, uncompressed video unless specified otherwise. These high bandwidth video streams also enable us to study performance issues associated with high-bandwidth multimedia applications.

In all tests described below, we measured jitter as follows: we timestamp each incoming video frame. The received interval is the time between the receipt of an entire video frame (last packet in a frame) and the receipt of the next entire frame. We computed the received intervals for a sufficiently large number of video frames (i.e. by running the live video over a long period of time). The average received interval was then calculated. We computed the
Fig. 4. Variation of maximum application jitter with cross-traffic network load using the three configurations in Fig. 2. CBR video was delivered using a UBR native ATM connection and cross-traffic network loads applied over a UBR native ATM connection.

Fig. 5. Variation of mean application jitter with cross-traffic network load using the three configurations in Fig. 2. CBR video was delivered using a CBR native ATM connection (with peak bandwidth reservation) and cross-traffic network loads applied over a UBR native ATM connection.
jitter as the difference between the average received interval and each individual received interval between consecutive video frames. The mean value of all the jitter values for all video frames was then calculated.

4. Performance results

4.1. Jitter performance

In the jitter experiments described below, we used live, uncompressed, 16-bit pixel, 640 × 480 video at 19 frames/s.

**Experiment 1.** We investigated the impact of ATM switch architectures with different buffering strategies on end-user application jitter. We experimented using three different configurations namely C1, C2, and C3 as given in Fig. 2. We investigated user jitter performance delivered over native ATM UBR and CBR native ATM connections. In this experiment, the network traffic load used was transmitted via another UBR native ATM connection. We used a 93 Mbits/s CBR video stream in this experiment.

**Results.** We note from Fig. 3 that the mean jitter when CBR video is transmitted over native ATM UBR connection is around 50–100 μs. The maximum jitter is around 2–4 ms (Fig. 4). From the results in Figs. 3 and 4, the different buffering strategies used by DEC and IBM switches do not really cause any significant differences in the end-to-end jitter performance under different network loads.

**Experiment 2.** In this experiment, the goal was to investigate the impact of end-to-end bandwidth reservation (using CBR circuits) on end-user application jitter performance. We used a 93 Mbits/s CBR video stream transmitted over a native ATM CBR connection.

**Results.** From Fig. 5, we observe that the average jitter with CBR native ATM connections using bandwidth reservation is lower than UBR native ATM connections. The mean jitter is around 25–50 μs and almost the same (2–4 ms) for the maximum jitter (Fig. 6). Moreover, in the case of the maximum jitter results (CBR connection), the standard deviation observed is two times better (around 90–150 μs) than the UBR connection (around 200–300 μs). The few occurrences of some peaks (for both DEC and IBM switches) in the mean jitter results of Fig. 5 make it difficult to conclude whether the DEC or the IBM scheduling implementation is better in giving the best jitter performance. Further experiments will be conducted in the future to investigate this issue. However, it is clear from Fig. 5 that the scheduling implementations in both switches give CBR traffic transported over CBR native ATM connections on average a lower jitter than UBR traffic. Another interesting point to note from Fig. 5 is that the average jitter with IBM switches only (configuration C3 from Fig. 2) is consistently lower than with DEC switches.
Fig. 7. Variation of mean application jitter of a 7 Mbits/s CBR video stream (transmitted over a CBR SVC connection) with cross-network native ATM and UDP/IP traffic (transmitted over UBR SVCs).

Fig. 8. Variation of mean application jitter of a 30 Mbits/s CBR video stream (transmitted over a CBR SVC connection) with cross-network native ATM and UDP/IP traffic (transmitted over UBR SVCs).
better (25 μs) than the results obtained from either configurations C1 or C2 in Fig. 2.

Experiment 3. A typical network environment supports a mixture of different types of applications (e.g. CBR, UBR, VBR) which often use different types of protocols such as TCP/IP, UDP/IP, and native ATM. The emergence of native ATM applications will co-exist side-by-side with legacy applications based on TCP/IP and UDP/IP using IP-over-ATM implementations. In this experiment, we investigated the effect of competing UDP/IP and native ATM traffic on the jitter performance of CBR video that was delivered using a native ATM UBR connection. Cross-traffic was transmitted over UBR connections. The experiments were repeated to explore the impact that various CBR video bit rates (7, 30 and 93 Mbits/s) have on jitter performance. In all these tests, we used a homogeneous network with two IBM switches (as given by configuration C3 in Fig. 2).

Results. From Fig. 7, we note that at low video bit rate (7 Mbits/s) competing native ATM traffic has almost the same effect on average jitter (around 80 μs) performance as UDP/IP cross-traffic. At higher video bit rates (30 and 93 Mbits/s) (Figs. 8 and 9), native ATM cross-traffic results in worse (i.e. higher) jitter performance than corresponding UDP/IP traffic loads. However, as Figs. 7–9 show, native ATM cross traffic causes less jitter variations than competing UDP/IP traffic at high video bit rates. In our tests, the average jitter variation observed with competing native ATM is around 50–100 μs even under high loads.

4.2. Packet loss results

Experiment 4. In this experiment, the video sender transmits 640 × 480 (16-bit pixel) uncompressed video frames at 19 frames/s (93 Mbits/s per video stream). Each video frame is segmented into packets of size 9180 bytes (the maximum allowed by the underlying ATM device driver for native ATM connections). This maximum transmission unit size results in 67 packets per video frame. We used a traffic shaper at the sender and constrained the sending throughput to 93 Mbits/s. We measured the number of packets dropped at the receiver when live video is transmitted using UDP/IP over ATM and native ATM connections. We repeated the experiments using configurations C1–C3 of Fig. 2. Our goals in this experiment were: to investigate the effect of ATM switch architectures on packet loss, to explore the effect of cross-traffic (native ATM and UDP/IP) loading on our CBR video stream delivered over native ATM, and finally to compare performance in terms of packet loss of video transmission over UDP/IP versus video over video transmission over native ATM.

Results. Above 50 Mbits/s cross-traffic network loading, packets are dropped by the ATM switch as the aggregate throughput of the outgoing traffic exceeds the output.
Fig. 10. Variation of video packet loss (at DEC switch) with cross-traffic network load using configuration C1 (DEC-to-IBM) in Fig. 2.

Fig. 11. Variation of video packet loss (at IBM switch) with cross-traffic network load using configuration C2 (IBM-to-DEC) in Fig. 2.
capacity of the link (running at OC-3 rate) connecting the two ATM switches. This results in packets being dropped by the first ATM switch. Packet loss performance for video over UDP/IP is better with DEC (Fig. 10) compared to the IBM switch (Fig. 11). However, with video delivery over native ATM connections packet loss is lower with IBM compared to the DEC switch (Figs. 10 and 11). With IBM switches only (Fig. 12), video transmission over native ATM gives the lowest packet loss (within 15%) compared to DEC which was around 23% under high network loads.

In all the tests described below, we investigate the impact of network bandwidth reservation on packet loss for live, uncompressed, $640 \times 480$, 16-bit pixel (CBR) video (at 93 Mbits/s per video stream) across a loaded ATM network. We used configuration C3 (IBM-to-IBM) in Fig. 2.

Experiment 5(a). We delivered live video using a UBR native ATM connection. We loaded the background network traffic using another UBR UDP/IP (over Classical IP) connection. In another test, we loaded the background traffic using a UBR native ATM connection instead of UDP/IP.

Results (5a). Table 1 shows the packets loss measured at the receiver in each test. As expected, in unloaded conditions, and up to a loading of 50 Mbits/s, very few packets are dropped. However, as the throughput of the background traffic exceeds 50 Mbits/s, the percentage packet loss increases as the aggregate throughput (live video and background traffic) exceeds the capacity of the ATM link.

<table>
<thead>
<tr>
<th>Network load (Mbits/s)</th>
<th>0</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loaded with UDP (UBR)</td>
<td>0.01</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
<td>3.51</td>
<td>10.39</td>
<td>14.88</td>
<td>20.58</td>
</tr>
<tr>
<td>Loaded with native ATM (UBR)</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
<td>0.01</td>
<td>0.01</td>
<td>5.19</td>
<td>10.85</td>
<td>14.75</td>
<td>21.92</td>
</tr>
<tr>
<td>CBR SVC over native ATM</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 1
Packet loss (%) at the receiver when delivering live, $640 \times 480$, 16-bit pixel, uncompressed video over native ATM under competitive network loads. Cross-network traffic was delivered over native ATM and UDP/IP UBR connections.
between the switches. At high competing network loads, Table 1 shows a slightly lower packet loss with competing UDP/IP traffic than with competing native ATM background traffic.

**Experiment 5(b).** We repeated the above experiment but using a CBR native ATM connection to deliver the live video instead of a UBR native ATM connection. The CBR native ATM connection used a peak bandwidth reservation of 93 Mbits/s. The end-to-end bandwidth reservation was made at connection time (including end-system and network switches) using ATM signaling and a traffic shaper at the end-system.

**Results (5b).** As given in Table 1, with network bandwidth reservation, packet loss was very low and almost constant (0.01%) throughout the experiments, even under heavily loaded conditions. This is in contrast to the case when a UBR connection was used to transmit the live video. This clearly demonstrates that with native ATM allowing end-to-end reservation of network resources such as bandwidth, the quality of service to end-user applications improves significantly. In addition, the results also show that CBR traffic is served with strict priority over UBR traffic in the switches. It was not possible to investigate how well other QoS parameters such as latency and jitter perform as the underlying ATM device driver does not support them yet.

**5. Conclusions**

In this work, we have experimented with live, CBR video traffic over both native ATM UBR and CBR connections. We placed emphasis on transmitting real-time, uncompressed (CBR) video over ATM connections through actual ATM switches in contrast to many previous studies and experiments based on theoretical simulations. We have focused particularly on the impact of competing traffic types and ATM switch architectures on jitter and packet loss performance in the presence of different types of competing cross-traffic (native ATM and UDP/IP) at different network loads. It is also worth noting that we have used standard native ATM implementations (e.g. standard Winsock 2 API with ATM support, the underlying ATM device driver supports UNI 3.1) in our experiments on native ATM compared to previous studies mainly based on IP-over-ATM networks or proprietary native ATM implementations. We summarize our major results below:

- Jitter performance over native ATM UBR connections appears to be independent of switch architectures with various competitive loading levels. Average jitter observed for a 93 Mbits/s CBR video stream is around 75 µs.
- With CBR native ATM connections, the scheduling implementations in the DEC and IBM switches ensure higher priority of CBR traffic over UBR traffic thereby minimizing the end-to-end jitter for CBR traffic. The average jitter obtained with both DEC and ATM switches for CBR live video is around 35 µs, almost half of that obtained with UBR native ATM connections (75 µs). There was no significant difference for the maximum jitter obtained over CBR or UBR native ATM connections. Further, we found that the IBM switches (with output buffering) yield better jitter performance than the DEC switch (input buffering with credit-based flow control). Based on the jitter results obtained, we note that the greater complexity of the DEC switch compared to the IBM 8265 ATM switch does not really yield better jitter performance.
- We found that at low video bit rate, the impact of competing UDP/IP and native ATM UBR traffic on CBR video jitter performance is almost the same. Our experimental results further show that at high bit rates (when using 30 Mbits/s and 93 Mbits/s live video streams), competing native ATM UBR traffic causes higher jitter than competing UDP/IP UBR traffic. However, we obtained smaller video jitter variations with competing native ATM traffic compared to the competing UDP/IP traffic.
- We also found that video jitter does not vary significantly with increasing CBR video bit rate when delivered over native ATM.
- We observed that packet loss depends on the switch architecture implementations. In our experiments, the DEC switch results in lower packet loss for UDP/IP UBR traffic compared to native ATM. In contrast, the IBM switch gives lower packet loss for native ATM compared to UDP/IP UBR traffic. Further, based on our experiments, we found that using a network with different types of switches (DEC and IBM in this work) causes higher packet loss than one with only one type of switch (in this case IBM).
- We showed that with end-to-end network bandwidth reservation over native ATM connections, we achieve very low packet loss with our CBR video application.

To conclude, we argue that jitter introduced by the network and switches on ATM local area networks is low and even under heavy network loads, the maximum jitter observed was still in the range of 2–4 ms. We believe like Partridge [29] argues it is not the responsibility of the network to control jitter because the network is assumed to provide sufficient bandwidth. Rather, as Partridge also argues, it is the operating system at the end-system that should take care of jitter for time-sensitive traffic [29]. In this context, in previous works [27,30], we have also demonstrated experimentally that jitter at the end-system suffers much greater variations than the results obtained in this work. More jitter control is needed at the end-system rather than the network. This work has also shown that end-to-end network bandwidth reservation over ATM networks.
greatly minimizes packet loss. Although network bandwidth reservation is now achievable with current ATM switches and some commercial ATM device drivers, much work remains to be carried out to achieve full end-to-end QoS support for other parameters such as latency and jitter. For instance, current commercial network switches and end-system ATM device drivers do not provide guarantees for these QoS parameters. One of our current areas of investigation is the provision of QoS guarantees, particularly at the end-system, for parameters such as delay and jitter. This requires careful scheduling and resource management implementations at the operating system and network interface levels. We argue that only with a combination of end-system and network QoS support will the full deployment of end-to-end QoS be practical.

Acknowledgements

This work has been supported by NSF grants EEC-9529152 and MRI-9724567. Equipment donations from the Intel Equipment Grant Program and the IBM SUR Grant are also gratefully acknowledged. The author also thanks Weiyou Cui for his help in all the experiments described in this paper. The author also thanks Xuan Chen and the anonymous reviewers for their thorough reviews and valuable comments that greatly improved the quality of this paper.

References