 Published in the Proceedings of the SPIE International Symposium on Technologies and Systems for Voice, Video, and Data Communications (Photonics East), Philadelphia, PA. USA, October 1995, pp. 256-264.

Local quality of service provisioning to meet end-to-end requirements in ATM networks

William Yurcik, *David Tipper* and *Sujata Banerjee*
Telecommunications Program
University of Pittsburgh
Pittsburgh, PA 15260

{yurcik,tipper,sujata}@tele.pitt.edu

ABSTRACT

Applications in high speed ATM networks will specify quality of service (QoS) requirements on an end-to-end basis. Typical application QoS metrics are cell loss rate, delay jitter, throughput, and maximum delay. Obviously, the individual network control components will need to provide more stringent quality of service than specified by the application. In this paper, we study (via simulation) how end-to-end QoS parameters provided by the application can be mapped into the QoS numbers used by the individual network control components.

1 INTRODUCTION

The architecture of high speed networks will consist of a high speed wide area backbone network, connecting smaller networks together. The rationale for such an architecture is that individual networks and users will require but a fraction of the capacity of a gigabit link (A video feed requires data rates of the order of Megabits per second.). The traffic from the smaller networks will be statistically multiplexed onto the high speed backbone using ATM (Asynchronous Transfer Mode: fast packet switching with fixed size packets or cells) technology. Traffic on the high speed ATM backbone network will be switched and routed by ATM switches. End-to-end virtual circuit connections between source and destination nodes will be set up. Applications will specify end-to-end QoS requirements to the network, which will be used by individual ATM switches to provide the required quality of service. Obviously, the individual ATM switches will need to provide a more stringent quality of service. For instance, if an application can tolerate only x units of end-to-end delay, then each individual switch can only cause the application packets to be delayed by a fraction of x.

Future high speed Asynchronous Transfer Mode (ATM)-based networks will provide stringent quality of service (QoS) guarantees to various applications.[1,2] These QoS parameters will be specified on an end-to-end basis by the application programs. Further, various intermediate network components will use the QoS values to reserve resources (bandwidth, buffer space, etc) and select virtual circuit routes such that the QoS parameters will be guaranteed without regard to future changes (e.g., traffic increases) in the network. While there has been considerable research in using given QoS parameter values to perform appropriate resource (bandwidth and buffer space) allocation in ATM networks, there is a lack of research efforts in the allocation of the end-to-end QoS required by an application to individual network components. In this paper, the problem of mapping the end-to-end QoS numbers to local QoS numbers is studied via simulation techniques.

One possible, although naive technique can be to assign a QoS number to the individual control mechanisms that is significantly more strict than the end-to-end requirement. While this method will often succeed in guaranteeing the end-to-end QoS requirement, extra resources such as bandwidth and buffer space will be expended, possibly resulting in the rejection of new connections whose QoS requirements can no longer be met by the network. The network may also suffer a lower utilization if the QoS numbers at the local network components is over-estimated. While some recent papers do address this problem, only simple cases are dealt with, using simple traffic models. Note that for a mapping to perform optimally, it must factor in the effects of the current network load and the fact that the application traffic characteristics will change while traversing through the network. These factors have been largely ignored in the current literature. Here we present results illustrating how the *local QoS* provided by the network components varies throughout the network, especially between the edge of the network and points deep within the network.

The problem of providing QoS guarantees to applications is very closely tied to the problem of connection (call) admission control (CAC). The CAC software will admit a new connection in to the network only if the QoS required by the connection can be guaranteed. Thus if the network overestimates the QoS requirements at individual switches for each connection, the CAC module will be forced to reject connections owing to insufficient resources to guarantee the QoS requirements of the incoming connections. Thus the network will be under-utilized. There have been several approaches to providing QoS guarantees that have been proposed in the last few years (See[1] for a good survey and discussion of these approaches.). Most approaches do not take in to account that traffic changes as it traverses network nodes. Those that do, can be classified into two extreme classes. Either the approach does not allow the traffic characteristics to change by using a fairly complicated queueing discipline that maintains timing information about each session flowing through that switch (e.g., the stop-and-go queueing mechanism), or, the approach develops worst case QoS bounds that are calculated on the assumption that traffic changes as it traverses the network. The first approach results in the local QoS allocation to be at the peak rate, while the second approach results in overly conservative (loose) bounds. Both approaches cause *overbooking* of resources, leading to poor utilization of the network. Thus, the characterization of traffic *deep* inside the network, and assigning local QoS parameters accurately is a crucial problem in the development of high speed wide area network technology. So far, the only work in this area known to us is for stationary traffic, simplistic models such as M/M/1/K and voice sources,[3] simulation studies for uniform traffic and simple ATM switch models,[4] and queueing analysis of ATM tandem queues.[5]

The paper is organized as follows. In section 2, the problem of QoS allocation is described in more detail. In section 3, the simulation model is described. The experiments and the results obtained are presented in section 4. Finally, some conclusions arrived at, and comments on future work are given in section 5.

2 AN END-TO-END QOS PROBLEM

Consider the problem of providing guarantees on the end-to-end delay jitter to a single virtual connection (VC) traversing several nodes in a network. We wish to determine whether given the traffic characteristics of the VC , the jitter requirements and current network loading, the VC can be supported by the network. Providing QoS requirements on the delay jitter is particularly important for constant bit rate (CBR) traffic. It is expected that CBR audio and video traffic will initially comprise a major portion of the traffic in wide area ATM networks.

The delay jitter is essentially a measure of the change in the time between consecutive cells of a connection (i.e., the change in the interarrival times). Hence, the jitter is a measure of the distortion introduced by statistical multiplexing in the network. Several different definitions for delay jitter have appeared in the literature. Here we follow the definition of delay jitter in[6,7] and assume that time is divided into slots of constant length equivalent to one cell transmission time on a link in the network. Let IA_k, $k \geq 1$ denote the interarrival time in slots at the traffic sourece between the $(k+1)$ cell and the kth cell of a VC. We define t_k^l to be the time slot that the kth cell arrives to monitoring point l in the network. Then the delay jitter at point l in the network DJ^l is defined by the sequence[8,7]

$$DJ_k^l = (t_{k+1}^l - t_k^l) - IA_k \qquad for\ k \geq 1$$

One can readily see that DJ_k^l measures the amount of distortion of the interarrival time IA_k at point l. The delay jitter DJ_k^l should be zero for no distortion to have occured. Note that the delay jitter as defined above can take on both positive and negative values. Positive values correspond to *cell spreading* where cells k and $(k+1)$ are monitored at a spacing greater than IA_k, whereas a negative delay jitter indicates *cell clustering* with the cells k and $(k+1)$ being monitored at a spacing less than IA_k. For CBR video and audio applications a small amount of either positive or negative delay jitter will not be noticeable to a human end user. In general end-to-end negative delay jitters are easily compensated for by buffering the cells at the end user and then delaying the playout of the $k+1$ cell until the correct interarrival time. However, it has been noted that a negative delay jitter at a point internal to a network can in effect result in a connection temporarily exceeding it's declared peak rate value within the network[1] and may require internal policing.[8] Large positive end-to-end delay jitters will induce distortion noticeable to the end user and are difficult to correct for. However, negative end-to-end delay jitters can be advantageous to smooth out the positive delay jitter values at the end user.[9] For example if the $k+1$ cell arriving results in a negative delay jitter, one can delay it's play out until a small positive delay jitter occurs if this results in a decrease in a larger positive delay jitter for the $k+2$ cell. An example of such an adaptive playout scheme for video connections is given in.[9]

The value of the end-to-end delay jitter incurred by a VC will depend on several factors such as the traffic characteristic of the VC, the background load in the network, the queue scheduling policies used, buffer management mechanisms, and number of nodes traversed. Obviously an exact analysis is difficult and several approximate analyses for the case of CBR VC traffic have appeared recently.[8,7] Simple approximate bounds can be determined on the delay jitter by modeling the path of the VC under study as a series of finite capacity FIFO queues. The minimum value for the delay jitter occurs when cells k and $k+1$ occupy consecutive slots and is simply $-(IA_k - 1)$. The maximum value for the delay jitter can be bounded by realizing that an increase in the delay jitter occurs at a queue in the VC path when the number in the queueing system seen by the $k+1$ cell is *larger* that seen by the kth cell. Consider queue 1 in the VC path with finite capacity N^1. Let MA_k^1 denote the maximum number of background traffic arrivals that could occur at queue 1 during time interval IA_k. The maximum change in the delay jitter mdj^1, that can occur at queue 1 with no lost cells is the maximum change that can occur in the number in the queueing system between two consecutive cell arrivals of the tagged VC. Specifically, $mdj^1 = min\{MA_k^1, N^1 - 1\}$. One can similarly determine mdj^l for the lth queue in the path by $mdj^l = min\{MA_k^l, N^l - 1\}$ where MA_k^l is determined by considering the worst case increase in the delay jitter at all upstream nodes to determine the time interval over which the maximum number of background arrivals is calculated. Hence, the following bounds on a the delay jitter (assuming no cells are lost due to buffer overflow) holds.

$$-(IA_k - 1) \leq DJ_k^l \leq \sum_{i=1}^{l} mdj^l$$

Notice that this bound assumes the worst possible case and as such is very conservative. Hence, using such a bound for admission control would lead to very low network utilization. In the following we look at how the delay jitter varies in a network and how this can be incorporated into QoS allocation.

3 SIMULATION MODEL

In this section, the simulation model of the system is described, as well as the experiments that were conducted. The basic unit of our simulation model is a 2x2 ATM switch based on the shared bus architecture[10] (See Figure 1) with output buffering. The shared bus architecture was chosen since it is one of the more commonly used implementations of ATM switches presently. In this architecture, a cell arriving to an input port of the switch is first converted from serial form into parallel bytes (or words). After the serial to parallel conversion, an internal header is attached for routing to the appropriate output port based on the VCI/VPI field of the cell. The cell

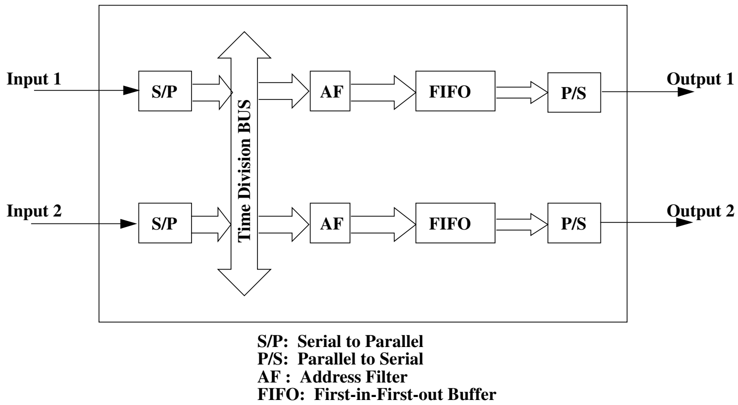

Figure 1: Shared bus architecture for a 2x2 ATM switch

is then placed in a queue to wait transmission on the shared bus. The bus uses a simple polling mechanism to service cells from the input ports and operates at a data rate higher then the input ports. The bus broadcast cells to all output ports where the address filter picks off (based on the internal header) the cells destined for that port. The address filter strips off the internal routing header and places the cell into a FIFO queue for parallel to serial conversion and transmission on the outgoing link.

Each aspect of the shared bus architecture was faithfully incorporated in the simulation model. All relevant parameters of the switch were calculated based on a DS-3 (45 Mbps) rate of incoming and outgoing links and a 64-bit data path for the bus. The queues on the input side of the bus were of size 2 cells and the output buffers were of size 64 cells. The incoming traffic distribution and the route of a tagged virtual circuit (VC) are input parameters of the simulation model. The simulation model was validated by conducting detailed traces of packets for deterministic arrival processes which could be checked analytically.

The overall system under study was a tagged VC traversing 3-nodes of 2x2 ATM switches, as shown in Figure 2. This system could be for instance, part of an ATM backbone network. The simulation was set up in a manner that background traffic could enter and leave at each of the switches or could be routed to downstream switches. However, the tagged VC that was monitored entered the first switch (switch 1) and was routed through switch 2 and switch 3, as shown in Figure 2. This route was fixed for all the experiments reported in this paper, in order to reduce the number of free variables.

All simulations were carried out in the SLAM-II simulation language. The standard method of replications with deletion of the initial transient was used to statistically analyze the simulation output data. Typically each simulation data point was the result of averaging 10 independent replications with each replication consisting of 30,000 or more VC cells being delivered end-to-end. The resulting confidence intervals were of small with typically a relative precision of less than 5% for a 90% confidence level. The confidence intervals are not displayed here for the sake of clarity in the graphs.

4 EXPERIMENTATION RESULTS AND DISCUSSION

The VC model of Figure 2 was studied for various sets of parameters and typical results are shown here. In each case, the main QoS parameter was the delay jitter of the tagged VC, which was measured at the output of each switch. In calculating the delay jitter dropped cells were ingnored. The tagged VC carried constant bit rate (CBR) traffic with one cell at the source every T slots. The average rate of the tagged VC is denoted by $\lambda_t = 1/T$. In order to model the effects of other traffic streams in the network interferring with the tagged VC traffic,

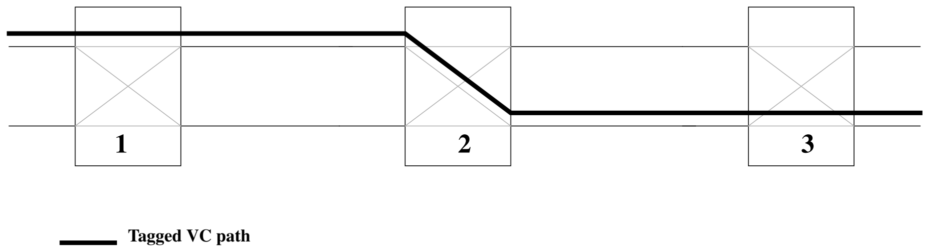

Figure 2: Three node model of VC

background traffic was generated at each switch input port. The background traffic was generated independently at each switch port according to a Bernoulli process with the parameter p_{ij} at switch i input port j. While the route for the tagged VC was kept fixed, each background traffic cell arriving to a switch input port is routed with equal probability to either of the two outgoing ports. This corresponds to a balanced loading scenario on the switch. Only part of the outgoing background traffic at a switch output port is routed to the next switch. The fraction of background traffic at switch i (i=1,2), output port j (j=1,2) that is routed to the next switch is denoted by l_{ij}. Thus, each experiment is defined by the following parameters: T, p_{ij} and l_{ij}.

One of the most widely proposed techniques for QoS allocation is to analyze a typical network node in isolation at the network edge and then utilize the results of this analysis to form end-to-end QoS decisions. Following this approach, the first experiment considered was a baseline scenario, with each node being essentially identical using the following parameters: $T = 8, p_{11} = p - 1/T, p_{12} = p, \{p_{ij} = p : i = 2, 3 : j = 1, 2\}, \{l_{ij} = 1, \ \ i = 1, 2 \ j = 1, 2\}$. The value of background traffic p was varied from 0 to 7/8 in increments of 1/8. Figures 3, 4, and 5 show the delay and delay jitter behavior. In Figure 3, the average delay of the VC traffic from the source to across switch 1, 2, and 3 is shown versus the background load p. Note that the switch 3 curve shown in Figure 3 corresponds to the average end-to-end delay. The average delay jitter at the output of each switch was found to be approximately 0 (slots) for the background load range considered. This is not surprising since the positive and negative delay jitters tend to average out. As noted by Landry[8] it is the variance in the delay jitter that is the principal concern. Figure 4 shows the standard deviation of the delay jitter σ for the tagged VC traffic at the output of each switch versus the background load p. As one would expect both the end-to-end transport delay and the delay jitter standard deviation increase with both the number of nodes traversed and the background load. Cell loss only occured in significant amounts at a background load of $p = 7/8$ and dropped cells were not included in the calculation of the average delay and delay jitter. Notice that at low loads the end-to-end delay is approximately 3 times the single switch delay. However for the delay jitter, at low loads (e.g., $p = 2/8$) the standard deviation of the delay jitter at the third node is greater than 3 times the standard deviation of the delay jitter at the first node. On the other hand at high levels of background traffic (e.g., $p = 6/8$) the end-to-end standard deviation of the delay jitter is much less than three time the standard deviation of the delay jitter at the first node.

In addition to the standard deviation of the delay jitter, the minimum and maximum values of the delay jitter are of interest. Figure 5 shows the extreme values (i.e., the minimum and maximum values) for the delay jitter observed at the output of each switch versus the background load p. Note that each value shown is obtained by simulating over 300,000 tagged VC cells and the curves marked switch 3 are the extreme values for the end-to-end delay jitter. From Figure 5 one can see that the extreme values for the delay jitter are symmetic about zero for low loads. Also, as the load increases the minimum value of the delay jitter encountered becomes -7 slots which corresponds to the cells from the VC occupying consecutive slots. As expected Figure 5 illustrates that the extreme values of the delay jitter increase with the number of nodes travesed and the level of the background load. The bounds given in Section 2 for the extreme values of the delay jitter when evaluated for the VC studied

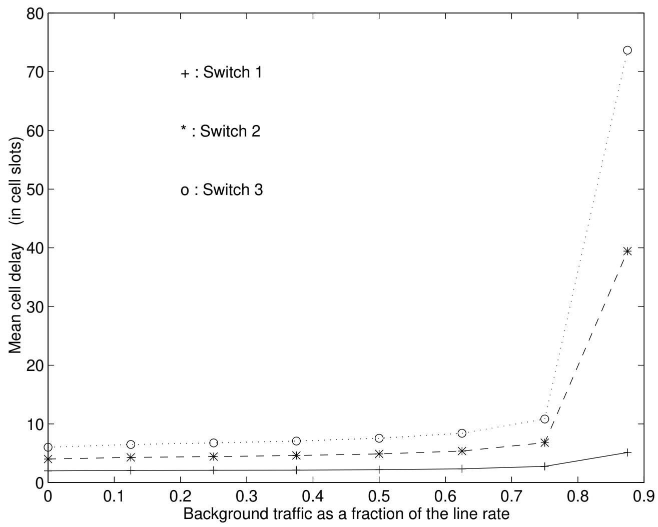

Figure 3: Tagged VC mean cell delay along path

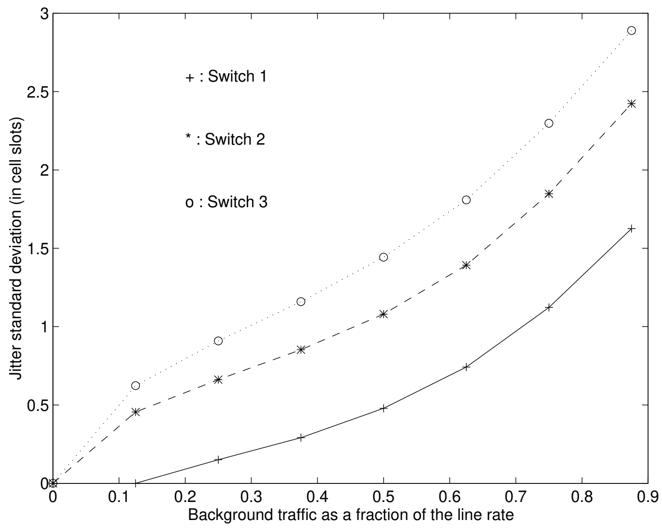

Figure 4: Average standard deviation of jitter for tagged virtual circuit traffic

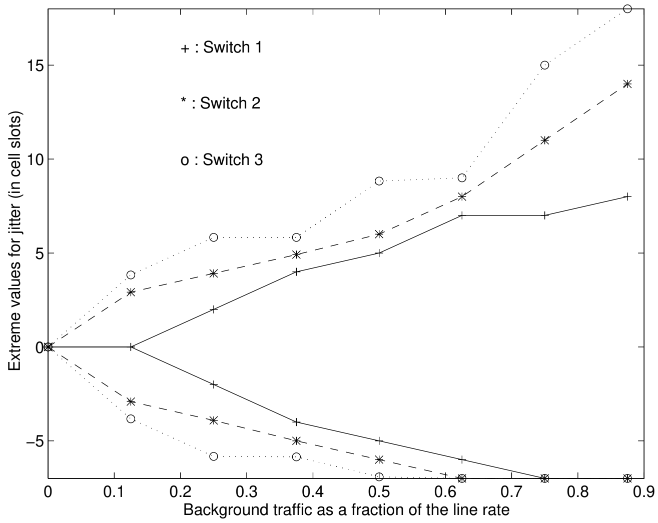

Figure 5: Maximum and minimum of jitter for tagged virtual circuit traffic

	Switch 1			Switch 2			Switch 3		
	σ	**Min**	**Max**	σ	**Min**	**Max**	σ	**Min**	**Max**
Case 1	1.12	-6	+7	1.25	-6	+7	1.61	-7	+8.8
Case 2	0.15	-3	+3	0.89	-4.91	+4.91	1.04	-5.83	+5.91
Case 3	1.12	-6	+7	1.60	-7	+9	1.92	-7	+11
Case 4	1.12	-6	+6	1.85	-7	+11	2.35	-7	+14

Table 1: Jitter metrics for unbalanced switch loads

here result in

$$-7 \leq DJ_k^1 \leq +8, \qquad -7 \leq DJ_k^2 \leq +24, \qquad -7 \leq DJ_k^3 \leq +48$$

Note that the maximum bound on the delay jitter is obtained only for the first node at heavy loads. This is part due to the fact that the bounds are determined on a worst case analysis and the probability of two cells from the VC finding each node in its worst case situation is very low.

The first experiment dealt with completely uniform traffic loadings at each of the three switches. Next, we consider 4 cases of unbalanced switch loadings. The average value of the standard deviation of the jitter (σ), the minimum jitter as well as the maximum jitter observed at the output of each switch, for these 4 cases are presented in Table 1, and the cases are labeled case 1–4. In the following, we discuss each case in detail.

As mentioned before, one of the major deficiencies in QoS allocation has been that changes in traffic as it traverses several nodes have not been taken into account while allocating local QoS. Logically, the switches that are heavily loaded should be assigned a looser local QoS constraint, while the lightly loaded switches must satisfy a stricter QoS requirement. However, if the changes in traffic characteristics are not taken into account, and only the traffic source model at the edge of the network is used, the load characteristics of switch 1 in Figure 4 would dictate the local QoS allocation for each switch according to the individual switch load. The results below (for each case) show that the above scheme causes the standard deviation of the jitter to overestimated sometimes, and underestimated at other times, leading to a poor local QoS allocation.

- Case 1: The overall background traffic rate entering switch 2 was lower than that at the other switches. The following parameters were set.

$T = 8, l_{ij} = 1,\ \ i = 1, 2\ \ j = 1, 2.$
$p_{11} = \frac{5}{8},\ p_{12} = \frac{6}{8}\quad p_{21} = \frac{1}{8},\ p_{22} = \frac{2}{8}\quad p_{31} = \frac{6}{8},\ p_{32} = \frac{5}{8}$

From table 1, it may be seen that average standard deviation of the jitter does not increase linearly with the load on the individual switches. If local QoS allocation were done according to the total load on the switches using the switch 1 graph in Figure 4, the standard deviation of the jitter allocated to switches 1, 2 and 3 would be 1.12, 0.15, and 1.12, respectively. Thus the standard deviation of the jitter at switch 2 would be 1.27 using this allocation scheme, but from table 1, we find this to be 1.25, an overestimation of 0.2. Similarly at switch 3, the standard deviation would be overestimated by 0.78.

- Case 2: The overall background traffic rate entering switch 2 was higher than that at the other switches. The following parameters were set.
  $T = 8, l_{ij} = 1,\ \ i = 1, 2,\ \ j = 1, 2.$
  $p_{11} = \frac{1}{8},\ p_{12} = \frac{2}{8}\quad p_{21} = \frac{3}{8},\ p_{22} = \frac{4}{8}\quad p_{31} = \frac{2}{8},\ p_{32} = \frac{1}{8}$

  Using Figure 4, the standard deviation of the jitter allocated to switches 1, 2 and 3 would be 0.15, 0.478, and 0.15, respectively. In this case, this allocation scheme underestimates the local QoS by 0.262 at both switches 1 and 2.

- Case 3: Unlike the above three experiments, part of the background traffic entering at switch 1 was routed all the way through switches 2 and 3. The following parameters were set.
  $T = 8, l_{1j} = 0.375,\ \ l_{2j} = 0.375, : i = 1, 2\ \ j = 1, 2.$
  $p_{11} = \frac{5}{8},\ p_{12} = \frac{6}{8}\quad p_{21} = \frac{2}{8},\ p_{22} = \frac{3}{8}\quad p_{31} = \frac{3}{8},\ p_{32} = \frac{2}{8}$

  Repeating the same calculations as above, the standard deviation of the jitter is overestimated at switches 2 and 3 by 0.262, and 0.684 respectively.

- Case 4: This case is very similar to case 3, except that the fraction of background traffic going through each of the three switches is higher than in case 3, increasing the correlation between background traffic at each switch. $l_{11} = l_{12} = 0.545,\ \ l_{21} = 0.375,\ \ l_{22} = 0.545,$
  $p_{11} = \frac{5}{8},\ p_{12} = \frac{6}{8}\quad p_{21} = \frac{2}{8},\ p_{22} = \frac{3}{8}\quad p_{31} = \frac{3}{8},\ p_{32} = \frac{2}{8}$

  Repeating the same calculations as above, the standard deviation of the jitter is overestimated at switches 2 and 3 by 0.39, and 1.01 respectively.

From the above results, it seems that allocating the standard deviation of the jitter locally according to the individual switch loadings and the source traffic model at the network edge, consistently overestimates the QoS allocation required at high loads. Further, strong correlation of background traffic between successive switches causes the overestimation to be even larger. Since it is at high levels of loading that the maximum conservation of resources is required, the changes in the traffic sources need to be characterized in more detail so, more accurate QoS allocation could be made.

5 CONCLUSIONS AND FUTURE WORK

In this paper, the problem of local allocation of end-to-end QoS requirements was studied by simulating a 3-node ATM network. The problem of allocating the delay jitter for a constant bit rate virtual channel was studied. It was shown that the common technique of basing QoS allocation on the analysis of a network edge node can lead to over allocation of network resources at heavy loads and under allocation at light loads.

Future work includes the validation of the results presented here, using actual ATM traffic measurements on the vBNS (Very High Speed Backbone Network Service). The vBNS is a wide area ATM network operating at 155 Mbps funded by the United States government through the National Science Foundation. Also, since changing the network configuration or the ATM switch parameters on the vBNS is difficult, we plan on using traffic measurements on the vBNS to run a trace driven simulation to further evaluate our local allocation scheme. Further, using the simulation testbed, the effect of various congestion control traffic policing, and buffer allocation strategies on the QoS allocation will be studied.

6 REFERENCES

[1] J. Kurose, "Open Issues and Challenges in Providing Quality of Service Guarantees in High Speed Networks," *ACM Computer Communications Review*, vol. 23, pp. 6–15, January 1993.

[2] D. Towsley, "Providing Quality of Service in Packet Switched Networks," in *Performance Evaluation of Computer and Communications Systems* (L. Donatiello and R. Nelson, eds.), pp. 560–586, Springer-Verlag, 1993.

[3] R. Nagarajan, J. Kurose, and D. Towsley, "Local Allocation of End-to-end Quality of Service in High Speed Networks," in *Proceedings of the IFIP Workshop on Performance Analysis of ATM Systems*, January 1993.

[4] D. Yates, J. Kurose, D. Towsley, and M. G. Hluchyj, "On Per-session End-to-end Delay Distributions and the Call Admission Problem for Real-time Applications with QOS Requirements," in *Proceedings of the ACM Sigcomm Conference*, 1993.

[5] F. Kamoun and M. M. Ali, "Queueing Analysis of ATM Tandem Queues with Correlated Arrivals," in *Proceedings of IEEE INFOCOM '95*, April 1995.

[6] J. Roberts and F. Guillemin, "Jitter in ATM Networks and its Impact on Peak Rate Enforcement," *Performance Evaluation*, vol. 16, November 1992.

[7] W. Matragi, K. Sohraby, and C. Bisdikian, "Jitter Calculus in ATM Networks: Multiple Node Case," in *Proceedings of IEEE INFOCOM '94*, June 1994.

[8] R. Landry and I. Stavrakakis, "Traffic Shaping of a Tagged Stream in an ATM Network: Approximate End-to-End Analysis," in *Proceedings of IEEE INFOCOM '95*, April 1995.

[9] S. Soman and S. Singh, "An Experimental Study of Video Conferenceing over the Internet," tech. rep., Department of Computer Science, University of South Carolina, 1995.

[10] F. A. Tobagi, "Fast Packet Switching Architectures for Broadband Integrated Services Digital Networks," *Proceedings of the IEEE*, vol. 78, pp. 133–167, January 1990.