Quality Of Service
in
ATM Networks

Domenico Ferrari¹ and Dinesh C. Verma¹
TR-90-064
December, 1990

Abstract

B-ISDN networks of the future will have to handle traffic with a wide range of traffic characteristics and performance requirements. In view of the high bandwidth of these networks and the relatively large propagation delays involved in wide-area B-ISDN networks, the performance requirements can only be provided by reserving resources to communicating clients at the connection establishment time. However, reservation mechanisms for heterogenous bursty traffic usually result in a rather poor utilization of network resources. In this paper, we propose a simple admission control criterion that can be used to reserve resources for bursty as well as smooth traffic with delay and loss sensitivities. Our scheme leads to a reasonable value of the maximum utilization of network bandwidth (about 40%) for delay sensitive traffic with moderate burstiness (peak-to-average bandwidth ratios of about 4), even under the worst possible conditions. Actual utilizations can be higher if there is smooth traffic or traffic which is not delay-sensitive. Our admission control algorithm uses a well-defined traffic specification scheme which is easy to enforce and verify, and able to accommodate arbitrary degrees of burstiness. Extensive simulation experiments failed to show that our admission control criterion are incorrect, in the sense that the quality of service requirements of the traffic was always met, even in the worst case. Moreover, the scheme is simple and feasible at the high speeds required of B-ISDN networks.

¹Computer Science Division, Department of Electrical Engineering and Computer Sciences, University of California and International Computer Science Institute, Berkeley, California, 94720 U.S.A.
1. Introduction

Computer communications of the near future are likely to have both low-delay high-bandwidth capabilities and low-delay high-bandwidth requirements. Applications, ranging from file transfers and electronic mail to packetized voice and remote graphical visualization of supercomputer results, will place diverse and stringent performance requirements on networks. The communication abstractions provided by present computer networks fail to provide any guarantees about the delay, loss rate or delay variation their clients may experience. With the rapid transition into the multi-media age, more and more applications will require such guarantees or some quality of service to be provided by the communication network.

These quality of service requirements can be met by providing communication abstractions that provide guarantees about network performance, like real-time channels [Ferr90] or flows [Zhan89]. However, providing these abstractions requires dynamic priority scheduling to be implemented in the switches, i.e. involves scheduling packets according to priorities calculated on a per-packet basis (like the virtual clock mechanisms of [Zhan89]), which is difficult to implement at high speeds. Thus, we need a scheme which preserves the essence of the real-time channel algorithms, i.e. which is able to provide performance guarantees, yet avoids the need for complicated scheduling mechanisms. In this paper, we propose some simple schemes for resource management, that allow us to achieve the same goal.

Quality of service guarantees can be provided either pro-actively or reactively [Lein88]. In reactive quality of service management schemes, the sources are allowed to send data in any fashion they choose. Resources are not allocated to individual connections. However, when the network manager senses that the quality of the service provided by the network is in danger, or likely to be in danger, it takes remedial action to correct the situation, possibly by requesting some sources to send data at a lower rate. However, in wide-area networks with high latencies, the network may not be able to react rapidly enough. Thus, pro-active techniques involving resource management at call establishment time need to be explored.

Network resource management at call establishment can be done within the following framework. At call setup, a user declares the burstiness characteristics of the call to the network using call control parameters. Based on the values of these parameters, the network manager allocates the necessary resources to the given connection to assure the quality required for packet transfer delay and packet loss. If there are insufficient network resources available, the call will be rejected. The network monitors the packet stream coming from the user to verify that the stream conforms to the parameter values declared at call establishment. It also maintains a scheduling policy which allows more important or delay-sensitive packets to be given priority over the less important packets.

We can identify four crucial components of this resource management architecture: (1) Specification: the method by which the burstiness and traffic characteristics of the communicating clients are specified; (2) Rate Control: the mechanism used by the network to ensure that this contract is being followed by the clients; (3) Scheduling policy: determines which packet is to be transmitted next wherever queueing of packets may occur; and (4) Admission Control: the set of rules which allocates resources and determines whether it is possible to accept a new connection. Any resource allocation architecture for wide-area B-ISDN networks must specify these components precisely. To the best of our knowledge, no existing work in the the area of resource

---

1 Similar ideas can be found also in [Ande88], [Paru89] and [Cone88].
allocation for ATM (Asynchronous Transfer Mode) networks [Hui88] [Ohni88] [Prot88] [Gall89] [Hari89] [Gole90] specifies all of these areas precisely.

In the next section, we describe the environment assumed for our studies. Section 3 sketches the specification and rate control components of our solution to the resource allocation problem in ATM networks. Section 4 sketches our scheduling and admission control policies. Finally, Section 5 presents some simulation results and conclusions are drawn in the last section.

2. ATM Environment

In an ATM network, all information to be transferred from the source to the destination is packed into fixed size cells, which are identified and switched by means of a label in the cell header. The term asynchronous refers to the fact that cells allocated to the same connection may exhibit an irregular recurrence pattern, as cells are filled according to the actual demand. All communication in an ATM network is connection-oriented, i.e., a connection needs to be established before data transmission can begin. While connections can be duplex in general, we will confine our attention in this paper to simplex (one-way) connections. Duplex connections can be handled as two simplex connections in opposite directions.

The delay through an ATM network is determined by three components: (1) the propagation delay: which depends on the distance between end-users, is independent of the type of switching node used and is by no means reducible; (2) the switching delay, which is usually fixed, implementation-dependent, but very low[Yeh87] and (3) a queueing delay. The queueing delay can be controlled by restricting the number of connections through the switch.

We are not considering the delays involved in cell assembly and fragmentation at the edges of the network. The focus of this paper is only on bounding the delays in the network. The work can, however, be extended to incorporate those delays as well.

The cells lost in an ATM network can be divided into three categories: those lost due to errors in transmission, those lost due to lack of buffers at an ATM switch, and those lost because they were delayed too much. Since errors in transmission cannot be controlled, and the cells lost because of delays can be accounted for by bounding delays, we restrict our attention to guaranteeing a maximum loss rate due to buffer overflows in switches. In order to provide a bound on the queueing delay and the buffer overflow probability in the switches, we use a simple model of an ATM switch (described in the next subsection).

As a target solution for broadband ISDN, ATM supports a wide range of voice, data and video services. Narrowband services require low bandwidth or bit rates up to 2 Mbits/second. Examples are audio applications (telephony, hi-fi sound, audio library and so on.) and short duration data (teletext, facsimile, telemetry, alarms, electronic mail). Broadband services require high bandwidth or bit-rates greater than 2 Mbits/second, e.g. video (video-telephony, broadcast TV, teleconference, pay TV, HDTV and so on.) and long duration data (word-processing, home-computing, database access, computer communication and so on.).

According to CCITT Rec.I.121, these services can be Continuous Bit stream Oriented (CBO) or cell oriented (bursty) depending on the way data is generated on the source side. A CBO source generates cells periodically at regular intervals. On the other hand, a bursty source intermittently generates a series of cells. Within the B-ISDN, CBO services require much higher quality of service than do voice communications. The quality of CBO services has to be about the same as that for existing circuit switching networks, i.e., no delay fluctuations, cell loss rate of $10^{-9}$, switching delay of 0.5 ms [CCITT G.142]. While it is difficult to guarantee no delay...
fluctuations in an ATM network, our scheme allows us to bound the delay within the limits acceptable to the client. Provision of performance guarantees to the two kinds of bearer services requires a scheme for resource allocation in an ATM framework.

Within the realm of bursty traffic itself, we can distinguish two types of traffic: (a) delay-sensitive bursty traffic, which requires a switching delay of about 1 ms per switch, with the probability of a packet being delayed beyond this bound less than $10^{-9}$; and (b) loss-sensitive traffic which ideally should not lose any packets because of buffer overflows at any switch. Of course, perfect reliability can only be obtained by means of acknowledgments and retransmissions. Thus we can classify our performance requirements into the following three types:

Class 0 Traffic: this traffic is CBO, which requires a delay of less than 0.5 ms per switching node and no packet losses due to buffer overflows.

Class 1 Traffic: this traffic is bursty, and requires a delay of less than 1.0 ms per switching node. Packet losses due to a missed delay bound or buffer overflows are less than $10^{-9}$.

Class 2 Traffic: this traffic is bursty, requires no delay bounds, but must have no packet losses due to buffer overflows.

All clients must specify which of these classes of service they require at the connection establishment time. Thus our problem is to devise a way to support this menu of QOS (Quality of Service) in the model of an ATM switch described next.

2.1. ATM switch

The ATM switch can be modeled as a set of input queues (incoming links) that are switched through an interconnection network to a set of output queues (outgoing links). Usually the only purpose of the input queues is cell/bit synchronization and clock recovery; so the input queue size is minimal. Nevertheless, it should be noted that statistical switching techniques will necessitate large input queue sizes in some cases. The output queues are needed since cells from different incoming links that are to be routed towards the same outgoing link may arrive concurrently.

A description of the different switch architectures and justification of the above model has been provided by Karol.[Karol87]. For ease of presentation, we will assume that (a) the ATM switch is internally non-blocking, (b) cells arriving on different input links and destined for different output links do not interfere with each other, (c) input queues are non-existent, and (d) framing details can be ignored. This allows us to concentrate on one output link and devise a scheme to bound the queueing delay at the output port.

An ATM network consists of a number of switches connected in an arbitrary topology. Each connection can be modeled as passing through a number of queueing servers, each server modeling the output link of an intermediate switch. We also assume that each switch is accompanied by a switch manager, which is responsible for accepting or rejecting connections through the switch. The switch manager should be running the algorithms and tests described later. When a connection is established, the switching fabric maps cells arriving along that connection from the input link to the appropriate output link. We are not concerned here with the exact details of this mapping mechanism.

In the next two sections, we describe a scheme that provides quality of service in an ATM network consisting of a mesh of the switches described above. As mentioned in Section 1, this would involve describing the four areas we referred to above, i.e., specification, rate control, scheduling and admission control.
3. Specification

The provision of performance guarantees by means of resource allocation can only be done efficiently if the clients specify their traffic characteristics and obey them when transmitting data. Any such specification should have the following properties:

1) Exactness: the specification must be exact, i.e., the network must be able to verify whether the client is obeying his promises or not. Thus, specifications of traffic by means of probability density functions [Wang90] or complex correlation coefficients [Guse90] should be ruled out since there is no known simple way to verify if the traffic is obeying the specification or not.

2) Flexibility: The specification must allow the clients to describe traffic with widely differing burstiness characteristics and bandwidth requirements.

3) Simplicity: the specification must be simple and concise. The advantages of a simple specification are obvious.

4) Maintainability: the traffic characteristics of any connection will change as we move along its path. In general, the burstiness of any packet stream tends to increase as we proceed towards the destination. Thus, the amount of resources allocated to a connection will tend to increase along the path. The network must be able to account for this change in traffic characteristics and ideally be able to preserve the same traffic characteristics from node to node.

Three simple traffic specifications have been proposed in the literature for admission control. We consider them one by one.

1) The average rate model [Gole90] [Gall89] specifies the average bandwidth required by the connection over an averaging interval. This could be looked upon as a window-based mechanism, where at most \( r \) packets can be sent in an averaging interval of length \( I \). Alternatively, we can look upon it as a rate-based specification, when the average spacing between any two packets is at least \( x_{\text{ave}} \) in every interval of length \( I \) (with the obvious equivalence \( x_{\text{ave}} = I/r \)). A slight variation of the scheme would specify that the spacing between any two packets be at least \( x_{\text{ave}} \).

The disadvantage of the average rate model is that it does not allow for burstiness in a client's traffic. Thus, this model must be augmented by models which allow for variable rates or bursts.

2) The average-rate burst-size model [Cruz87] specifies a maximum burst-size in addition to an average rate. In every time interval of length \( I \), the number of packets on the connection must not exceed \( I/x_{\text{ave}} + b_{\text{max}} \), where \( x_{\text{ave}} \) is the average spacing between the packets and \( b_{\text{max}} \) is the maximum burst size. An intuitive explanation of the model is that it allows up to \( b_{\text{max}} \) packets to be transmitted in rapid succession, but maintains a long-term average rate of \( 1/x_{\text{ave}} \).²

This model allows some burstiness, reflected by the burst-size parameter. However, the burstiness (the burst size \( b_{\text{max}} \)) increases as one moves across the nodes in a network. In terms of resource allocation (using the resource allocation schemes of [Andr89]), this model tends to tax the nodes downstream much more heavily than the nodes upstream. We do not know of a simple way to maintain the value of the burst size parameter across the network.

² The actual specification by Cruz uses different terms, but is functionally equivalent to our description. We have attempted to use the same terms and symbols in the specification of all the three models.
(3) The peak-rate average-rate model [Ferr90] allows the sender to transmit at two different rates, a peak and an average rate. The minimum spacing between any two packets must be larger than or equal to $x_{\text{min}}$ at all times (this corresponds to the peak rate), while the average packet spacing in any interval of length $l$ must be larger than or equal to $x_{\text{ave}}$. In this model, burstiness may be expressed as the ratio of peak to average rate or $x_{\text{ave}}/x_{\text{min}}$ [Choi89].

This model allows burstiness to be modeled, and has the advantage that the minimum and average spacings can be maintained at all nodes along the connection's path by a simple rate control mechanism. This has obvious benefits for resource allocation schemes at the nodes downstream.

In general, the traffic generated by the clients in a B-ISDN network will not be well-behaved, and a leaky bucket mechanism [Turn86] at the entrance of the network will be needed to convert the natural unregulated traffic into the regulated traffic adhering to the above mentioned models. The efficiency of such leaky bucket schemes is discussed in Section 5. From those simulations, it seems that the peak-rate average-rate model is well-suited for specifying the traffic characteristic of connections in an ATM network at reasonable burstiness values, typically less than 5, and using an $l$ about twenty times $x_{\text{ave}}$ or longer.

Along with the traffic specifications, the client desiring to communicate must also declare his performance requirements. In our model, he does so by declaring which of the three classes of service in Section 2 he belongs to.

For ease of presentation, we assume that the averaging interval $l$ is a network-wide constant. It is possible to extend our resource allocation schemes to allow different averaging intervals for different connections, but the advantages of such extensions do not seem to be significant.

3.1. Rate control

Enforcement of traffic characteristics is a must since a malicious user could send cells into the network at a much higher rate than the declared maximum or average value. The same effect might be caused by a failure in the sending host or in a switch. Network load fluctuations can also cause the minimum inter-packet spacing along a connection to be less than the declared bound. If we do not take appropriate countermeasures, such malicious or faulty behavior can prevent the satisfaction of the delay bounds guaranteed to other clients of the real-time service, thereby damaging the clients and destroying the credibility of the service.

The enforcement of the traffic specification can be termed rate control. Rate control can be done by forcing the "offending" cells to wait till the time they were actually supposed to arrive. When buffer space is limited, packets that arrive too much in advance of their proper time might even be dropped because of buffer overflow. Of course, a sufficient amount of buffer space will have to be allocated to prevent the offending cells from flooding the buffer space of a heavily loaded switch and causing cells from other connections to be dropped.

Rate control needs to be done at each output queue of an ATM switch. Thus, the output queue consists of two modules, a regulator, which acts as a policeman preventing misuse of the network, and a scheduler, which is responsible for selecting the cells to be transferred next on the output link. (See Figure 1.)

- Thus, rate control requires that we calculate the expected arrival time of each packet along a connection, i.e., the time the packet should have arrived if it had obeyed its $x_{\text{min}}$ and $x_{\text{ave}}$ constraints. Calculation of this expected arrival time is easy and can be done by means of the simple
Figure 1. This figure shows the structure of the output link we propose for an ATM switch. The regulator is responsible for rate control. The scheduler is responsible for shipping cells according to their service requirements.

pseudo-code in Algorithm 1. (adopted from [Ferr89].)

On the acceptance of a connection, three kinds of information (the state) needs to be kept at each switch. The state information predicts the earliest time the next packet on a connection can arrive. The information can be initialized by means of a routine like initialize().

On the arrival of every packet the procedure packet_arrival() is called; the procedure produces the time the packet should actually have arrived at the node, if it had obeyed all the traffic specifications. If the packet happens to have arrived before it is expected, it is held for a period of time equaling the difference between expected arrival time and the current time, before being handed over to the scheduler.

The hardware implementation of the rate control scheme may be done by means of a number of calendar queues[Brown88]. For the time being, assume that we have an infinite number of such calendar queues. Calendar queue i would contain a list of the packets that were expected at time i, but may have arrived early. When the clock reaches the values i, all these packets are transferred to the scheduler by appending this list to the list of packets in the scheduler queue. The operation can be done in a constant number of steps. Since, the computation of the expected arrival times is done so as to preserve the $x_{min}$ and $x_{ave}$ constraints, the two properties are enforced at every switch along a connection's path.

In a practical implementation, we can only have a finite number of calendar queues. The effect of infinite calendar queues can be mimicked closely if the number of such queues is large enough to keep all the packets that would arrive in the current interval of length t. Suppose there are N calendar queues. Also assume that the time axis is discrete and divided into slots of length t, the time required to transmit one ATM cell on the output link. Then, we need to ensure that $Nt \geq t$. Packets that arrive too far in the future (more than N time slots ahead in the given
Algorithm 1

initialize()
{
    count = 0;
    expected_time = clock;
    next_I = clock + $I$;
}

packet_arrival()
{
    count++;
    if (clock > expected_time) expected_time = clock;
    answer = expected_time;

    /* Compute the expected time for the next packet */
    if (count >= $I/x_{ave}$) {
        expected_time = max(next_I, answer + $x_{min}$);
        count = 0;
        next_I += expected_time + $I$;
    } else {
        expected_time += $x_{min}$;
    }
}
return(answer);

example) would be dropped. As the clock advances, calendar queues which become empty can be used for holding the packets expected later in a wrap-around fashion.

The schematic diagram of rate control can be seen in Figure 2. Note that the rate control algorithm has to be performed at each switch to ensure that the traffic specifications are always satisfied. Also note that all the operations involved in rate control can be done in a constant number of steps. Since there is no list processing or priority queue manipulation, the rate control scheme can be implemented very easily in hardware.\(^3\)

4. Scheduling and Admission Control

Keeping in mind that the delay requirements of the CBO connections are more stringent than the delay requirements of bursty traffic, the first choice of a scheduling scheme appears to be one that gives priority to the CBO traffic over the bursty ones. Such a scheme is proposed by Murase et. al. [Mura89], where the authors show that it minimizes the influence of the bursty traffic on the CBO traffic. Since bursty traffic itself consists of two categories, the delay-sensitive bursty traffic must be given preference over other kinds of bursty traffic.

\(^3\) The rate control process has some similarities with a rate-based scheduling mechanism referenced in [Kana90].
In each ATM switch, three queues are maintained at each output port: one for cells belonging to each class. When the node must choose the next cell to be transmitted, the scheduler picks a class 0 cell, if one is present. Otherwise any existing class 1 cell is transmitted. Class 2 cells are transmitted only when no cells from any of the upper classes are present. This scheduling mechanism results in good delay characteristics for delay-sensitive traffic [Todo90].

This is a simple priority scheduling mechanism, which can be done at the high speeds required for B-ISDN networks. Certainly, this is not the optimum scheduling scheme for an ATM network; complex algorithms will in general perform much better [Fitz89][Chen89]. However, these algorithms may not be feasible at higher speeds.

4.1. The admission control tests

When a switch receives an establishment request message, it performs some or all of the following tests:

- the class 0 bandwidth test, required when the connection to be established belongs to class 0, and involving the existing class 0 connections sharing the same output link;
- the class 1 bandwidth test, to be performed for the establishment of both class 0 and class 1 connections if the new connection belongs to class 1, or if at least one class 1 connection shares the same output link; this test involves all class 0 and class 1 connections on the output link;
- the class 2 bandwidth test, to be performed in all conditions;
- the delay bound tests for class 0 and 1 verify if the delay constraints of class 0 and class 1 traffic are being met.
the buffer space test, which is needed for all types of connections in order to determine if sufficient buffer space is present in the switch to accommodate the new connection.

If the request passes the tests, then the switch manager sends the establishment message on to the next switch. Otherwise the request is rejected.

The class 0 bandwidth test consists of verifying that the outgoing link is fast enough to accommodate the additional class 0 connection without impairing the guarantees given to the others. Keeping in mind that class 0 connections have a constant bandwidth requirement \( x_{\text{min}} = x_{\text{ave}} \), we simply need to check that

\[
\sum_{j} t_{i}/x_{\text{min},j} < 1, \tag{1}
\]

where the sum is taken over all the class 0 connections sharing the output link with the incoming request. We would like to remind the reader that \( t \) is the time to transmit one cell on the output link.

The class 1 bandwidth test verifies if it is safe to statistically multiplex a number of bursty connections to achieve better switch utilization. It is responsible for determining whether the probabilistic delay guarantees for bursty connections already accepted by the switch are safe (within the bound of \( 10^{-9} \)) as required by the QoS guarantees.

Let us divide the time axis into regions where the bandwidth demanded by all the active connections is less than the bandwidth of the outgoing link, and into regions where it is not. When the bandwidth requested is more than the link’s bandwidth, we are operating in an unsafe region. Define the overflow probability as the probability that we are likely to enter such a region. An upper bound on the probability that a class 1 connection will be delayed beyond its delay bound is given by this overflow probability.

The probability that connection \( j \) is active (i.e., is carrying packets) at some instant of time during an interval \( t \) is

\[
p_{j} = x_{\text{min},j}/x_{\text{ave},j}. \tag{2}
\]

Given \( K \) independent connections passing through a node, the probability that the members of a subset \( C \) of them are simultaneously active is given by

\[
\text{Prob}(C) = \prod_{i \in C} p_{i} \prod_{j \notin C} (1 - p_{j}). \tag{3}
\]

Let at least one of the connections be a class 1 connection. To compute \( P_{o} \), the overflow probability at the node, we start by listing all the overflow combinations. An overflow combination is a set of connections that, when simultaneously active requires a bandwidth larger than that of the outgoing link. In other words, an overflow combination is one for which inequality (1) above is not satisfied.

The class 1 bandwidth test consists of computing this probability and verifying that the value does not exceed the threshold required by class 1 service. In other words, the overflow probability should remain less than the value of \( 10^{-9} \) required for class 1 grade of service. Although a brute-force evaluation of this probability will take exponential time, a good approximation can be obtained in a constant amount of time (if the accuracy required is fixed). This method is described in appendix 1.

The delay bound tests verify that the delay guarantees of the class 0 and class 1 connections can be met whenever the switch is not overloaded. If there are \( N_{0} \) connections which belong to
class 0 grade of service, then equation (1) implies that the maximum delay of a class 0 cell can not exceed \( d_0 \) [Cid88] where

\[
d_0 = (N_0 + 1) t,
\]

where \( t \) is the basic cell transmission time on the output link.

Let us consider the largest combination of class 0 and class 1 connections that satisfies condition (1). This set will consist of all the \( N_0 \) class 0 connections on the output link and some \( N_1 \) low delay burst mode connections, selected in the order of decreasing \( x_{\text{min}} \) values. The maximum delay of any class 1 cell, when operating in the safe zone of the time axis, is bounded above by \( d_1 \), given by

\[
d_1 = \left( \frac{N_0 + N_1}{(1 - \rho_0)} \right) t + t,
\]

where \( \rho_0 \) is the utilization of the output pipe by all the existing class 0 connections. In other words, \( \rho_0 = \sum i / x_{\text{ave},i} \), with \( i \) ranging over all the class 0 connections.

The delay bound tests consists of verifying that both \( d_0 \) and \( d_1 \) are within the acceptable limits of their respective grades of service. Thus \( d_0 \) must be less than 0.5 ms and \( d_1 \) must be less than 1.0 ms.

Buffer allocation is required in order to guarantee connections with a bounded loss rate. It is also known that a shared buffer architecture results in a much better buffer utilization than an architecture in which each connection is statically allocated a fixed number of buffers. Notice that the rate control scheme prevents greedy clients from usurping the whole of the shared buffer space. Thus it is safe to have connections share a common buffer space.

What is the total buffer requirement of all the connections sharing the same communication link? Let us look at the buffer requirements of the scheduler first. Let us examine the state of the scheduler at a time \( T \) (measured in numbers of the cell transmission time \( t \)) after it makes a transition from an idle to a busy state; the count of cells present in the scheduler queue can be bounded above by the amount \( C_T \) given by

\[
C_T = \sum_{i} \left[ T / x_{\text{min},i} \right] \left[ I / x_{\text{ave}} \right] - T,
\]

where \( i \) ranges over all the connections sharing the output link including the connection being established. It can also be shown that the maximum value of \( C_T \) will occur either at \( T = 0 \) or at \( T = \lceil I / x_{\text{ave},i} \rceil \), where \( i \) could be any of the connections sharing the output link. Thus, with \( N \) connections sharing an output link, the maximum buffer requirement (in number of cells) can be computed by obtaining the maximum value of equation (6) evaluated at the \( N+1 \) values of \( T \).

Keeping in mind the fact that the maximum number of cells a connection can have in the regulator is the maximum number of cells that are present in the scheduler immediately before the node, it follows that, in a network of homogenous switches, the buffer requirement of the regulator is the same as the buffer requirements of the scheduler. Thus, the buffer space test consists of evaluating the maximum value of equation (6) at the \( N+1 \) values of \( T \) mentioned in the previous paragraph, doubling this value to account for the regulator, and then verifying that the buffer space in the switch exceeds this computed amount. Of course, this test assumes that the regulator and the scheduler share the same buffer space as well.

These admission control criteria are simple and easy to enforce at connection establishment time with limited amounts of processing power. However, the astute reader will have noted that
both the delay and buffer computations are done on the basis of a worst-case assumption, assuming that all connections may be active at the same time. There is a cost to worst-case design in terms of poor network utilization; however, it is the only safe way if we are to ensure that we meet the required quality of service.

5. The Simulations

In this section, we present the results of the simulation experiments we ran to study the performance of our admission control algorithms. We were interested in a number of issues:

(1) Is the scheme correct? Can we discover cases in which the admission control schemes would fail to produce the desired results?

(2) Is the specification model used satisfactory? Can reasonable values of \( I \) be obtained? Can irregular traffic be converted easily to regular traffic? What are the typical burstinesses required for this conversion?

(3) How close are the predicted delay bounds and buffer requirements to the actual delays experienced and buffers utilized in our simulation experiments?

(4) What are the typical bandwidth utilization in the case of bursty traffic? Are these values of utilization acceptable?

![Figure 3. The five probability distributions examined in our leaky bucket studies. The distributions are exponential (Poisson), the hyper-exponential, the uniform, and the Gaussian. The other distribution consists of auto-correlated Poisson process.](image)

We obtained answers to these questions by means of a simulator written in CSIM [Schw87]. We measured the actual delays and buffer requirements observed in the simulations in the presence of all three kinds of traffic, and compared them against the predicted upper bounds.
given by our admission control rules. We were not able to discover any cases in which the desired quality of service was not met. While this can not be cited as a proof of correctness of our admission control scheme, it leads confidence that the schemes will work in all practical situations.

In order to see if the specification can be generally used, we tried to convert random traffic into the \((x_{\text{min}}, x_{\text{ave}})\) paradigm by means of a leaky bucket mechanism. Our bucket leaked at two rates, one corresponding to \(x_{\text{min}}\) for short periods, while the other one ensured that the average leak obeyed the \(x_{\text{ave}}\) bound. We examined a number of different arrival processes, whose probability distributions are shown in Figure 3. These distributions included the exponential, the hyper-exponential and the gaussian distributions. Some of these patterns also had autocorrelations in the inter-arrival times. Figure 4 shows the simulation results of such a leaky bucket for the exponential distribution. All delays and intervals in Figure 3 and 4 have been normalized by dividing with \(x_{\text{ave}}\). A value of about 20 arrivals (i.e. \(l=20x_{\text{ave}}\)) seems adequate for such a traffic. Also, the maximum delay in the leaky bucket with a burstiness (measured as peak to average bandwidth ratio) of 3 or 4 is less than one inter-arrival time \((x_{\text{ave}})\). We found the same to hold true of the other distributions, including distributions with auto-correlated arrival processes. For space reasons, we are not showing the results for the other distributions in this paper.

While these results do not cover the entire range of arrival distributions that one might experience in an ATM network, they suggest that a lot of traffic on the network can be modeled by our specification scheme with moderate values of burstiness. Of course, there will be other processes which may have burstiness of up to 10 or more, but they can also be modeled well by our scheme.

In order to answer question (3), we present the results of a typical run of the simulator. Connections were created with a peak bandwidth 1/20 of the output link bandwidth, and bursty connections had a burstiness of 4. A connection was equally likely to be of any class. An on/off model was used for generating bursty traffic, and the network wide \(l\) was assumed to be 400. In order to keep the simulation runs to reasonable lengths, we observed the amount of delay we had in our simulations at different probabilities up to a probability of \(10^{-4}\) and extrapolated it to \(10^{-9}\). Figure 5 compares the simulated delays with the predicted maximum delays, and Figure 6 compares the buffer allocation with the actual utilized buffers. Our predicted bounds are always met, confirming the correctness of the admission control schemes. The bounds appear to be reasonably close to the observed values, yet far enough as to be safe.

In order to test the behavior of our admission control policy in the face of bursty traffic, we assumed the worst possible situation for our admission control policies. Thus, all connections were assumed to be of class 1, which would lead to the lowest possible utilization of the network. The delay bound \(d_1\) was set to be equal to 100 times the time required to transmit a cell on the output link; this roughly corresponds to a delay of 1 ms on a 45 Mbps network. Figure 7 shows the maximum possible utilization of the output link in the presence of only class 1 connections, with the probability of a cell missing the deadline of \(d_1\) equal to \(10^{-9}\). On the horizontal axis, we have the minimum inter-arrival time of a single connection, i.e. \(x_{\text{min}}\). The figure shows that utilization is poor for bursty connections with high peak bandwidth, but fair for connections with lower bandwidth, less than 1/30 of the total output link bandwidth. From Figure 7, we see that utilization when the peak bandwidth is 1/40 of the output link bandwidth, is about 40% with a burstiness of 5. Thus, a 45 Mbps output link can support traffic with burstiness of 5 at a utilization of 40% if the maximum peak bandwidth is about 1 Mbps. \(^4\)

\(^4\) This is a worst-case situation, when all the traffic is bursty. Even in this case, we can support higher peak bandwidths efficiently if we permit a higher overflow probability, e.g. at the order of \(10^{-6}\), traffic of burstiness 4
Figure 4. The delays involved in converting a Poisson (unregulated) packet stream to the peak-rate average-rate model. Burstiness is measured by the ratio of the peak and average rates. On the vertical axis, we have the maximum delay experienced by a packet in the leaky bucket. This delay shows a sharp decline as we increase the burstiness. On the horizontal axis, we have the I parameter of the specification model. Both the axes have been normalized by dividing them by the average inter-arrival interval. A burstiness of about 4 and an I of about 20 times the average inter-arrival time seems sufficient to convert the process without large delays.

would be higher if class 0 and class 1 grade connections were present as well. Since operational utilization of ATM switches are likely to be between 40% and 70%, delay sensitive bursty traffic would compose about a third of the traffic, a scheme that can support 30% utilization considering only bursty traffic can be deemed reasonable. Figure 7 shows that our scheme can accommodate peak-bandwidths about 1/30 of the maximum bandwidth at a burstiness of 5 even in the worst case situation described above. Thus, our scheme seems to work reasonably well.

As a final note, Figure 7 discusses only bursty traffic with homogenous characteristics; We also examined a large number of cases in which the burstiness and peak bandwidth were varied randomly. The mean utilization in the presence of delay sensitive bursty traffic alone was found to be about 46% with a standard deviation of 20%.
Figure 5. The delay bounds computed by our admission control policies, and the actual delays experienced by the cells at different level of utilization. The average burstiness for class 1 traffic was about 4. Notice that the curves become flat after some values of utilization. This occurs when our admission control schemes would admit no more class 0 or class 1 connections.

6. Conclusions

This study was undertaken to investigate the feasibility of pro-active resource allocation in order to ensure the quality of service in an ATM network. We defined the problem, offered a menu of quality of service options, and described the components of a solution that implements the classes of service offered by our menu. We examined the feasibility of a simple specification scheme, and devised a rate control mechanism to enforce strict adherence to this specification limits.

In order to provide our classes of services, we proposed a very simple scheduling mechanism for ATM cells at each switch, and devised resource allocation and admission control rules that are appropriate for the specification and the scheduling mechanisms. Our results have been satisfactory; the bandwidth of the network is used relatively efficiently, arbitrary degrees of burstiness can be accommodated and channels can be granted arbitrary amounts of bandwidth on demand. All the steps in our solution are simple and can be implemented easily in hardware at high speeds.

7. References

Figure 6. The buffers allocated to a switch's connections and the actual usage of buffer space in this switch. The bounds are pretty crude at lower utilizations, but appear to be moving closely at higher values of utilization. Only the scheduler buffers have been shown in this figure.

376-383.


Figure 7. The maximum possible utilization of the switch when all the traffic is of class 1. The horizontal axis shows the minimum inter-arrival interval of the traffic while the vertical axis shows the utilization under those conditions. Five degrees of burstiness from 1-5 were explored. These can be compared with the maximum possible utilizations at burstiness of 10 and 100. In general, burstiness in traffic reduces the maximum possible utilization, but the decrease is more significant at low burstiness values than at higher burstiness values. Utilization generally improves when we are multiplexing a number of low bandwidth channels; the gain by multiplexing is not much at higher peak bandwidths.


[Gole90] S. J. Golestanis "Congestion-Free Communication in Broadband Packet Networks", Proc. ICC 90, pp 308.3.1-308.3.6


[Hari89] B. Harita and I. M. Leslie, "Dynamic Bandwidth Management of primary rate ISDN to support ATM access", SIGCOMM-89, pp 197-211.


Appendix 1

A fast approximation for overflow probability

The problem of evaluating the overflow probability is identical to the following probability problem:

Given $N$ independent variables $X_1, X_2, \cdots, X_N, X_i$ taking the value $w_i$ with probability $p_i$, and the value 0 with probability $1 - p_i$, compute the probability that $X_1 + \cdots + X_N \geq 1$.

Computation of the overflow probability can be done by evaluating this probability, defining a variable $X_j$ for each of the channels existing at a node, with the values:

$$p_j = \frac{x_{\min,j}}{x_{\text{ave},j}}$$

$$w_j = \frac{t_j}{x_{\min,j}}$$

Figure 1

![Figure 1](image)

Figure 1. This figure shows the typical shape of the tail distribution of the sum of overflow probabilities. It consists of a number of steps of different sizes, beginning at arbitrary locations. The dashed lines show an approximation that bounds the tail distribution using steps of constant sizes.

Although we are required to find out only one value in the probability distribution, of the variable $Y = X_1 + \cdots + X_N$, namely the probability that $Y$ exceeds 1, we would determine the whole distribution. Keeping the whole distribution allows us to add channels or to delete
channels in a easy fashion. Assuming that we define the function \( f(x) = P_r(Y \geq x) \), the plot of \( f(x) \) would look like a series of steps as shown in Figure 1. The dotted lines show where the steps begin and end. In general, the length of the steps can be arbitrary and very small. In the worst case, there may be as many as \( 2^N \) steps in this functions, and maintaining the distribution will take \( O(2^N) \) time, when adding or deleting any channel. However, a reasonably good approximation can be made if we restrict ourselves to some \( K \) steps, and ensuring that the approximation thus obtained is an upper bound on the actual distribution. The desired approximation can be seen in Figure 1. The new steps are shown by means of the dashed lines. The bound can be brought closer to the real function by increasing \( K \). Also, closer bounds will be obtained if the approximated steps start at the same locations as the actual function steps.

The idea behind the fast approximation is to maintain a constant number of steps. Let us define the approximate function \( g(N,x) \) as the value of the bound when \( N \) channels exist at a node. Also assume that each of the steps in our approximation is \( \delta \) units long. ON adding a new channel \( N+1 \), we round off its weight to the next higher multiple of \( \delta \) and try to add this channel to the existing function. Conditioning on the new channel,

\[
g(N+1,\delta m) = g(N,\delta m)(1-p_{N+1}) + g(N,\delta (m-w_{N+1}/\delta))p_{N+1}
\]

(3)

Similarly, deletion of a channel (say \( N+1 \)) can be done by noticing that

\[
g(N,\delta m) = \frac{1}{1/(1.0-P_{N+1})}g(N+1,\delta m)-p_{N+1}g(N,\delta (m-w_{N+1}/\delta))
\]

(4)

Expanding the second factor on the right hand side repeatedly, and recalling that \( g(N,x) \) for negative or zero \( x \) is 1.0, we obtain that

\[
g(N,\delta m) = \sum_{k=0}^{m/w_{N+1}/\delta} [p_{N+1}/(1.0-P_{N+1})]^k g(N+1,m-k\delta/w_{N+1}/\delta)
\]

(5)

The advantage of using equation (5) rather than (4) in deleting a channel is that errors due to rounding off do not accumulate.

We now illustrate the approximation process by means of an example. Let us assume that there are 3 channels that are requested at the node. Let them have the following properties.

<table>
<thead>
<tr>
<th>Channel</th>
<th>p</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

For the sake of illustration, we choose \( \delta \) to be 0.2. This would require that we have six columns in our table. In the beginning the table is initialized so that the first entry is 1.0 and all the rest all zero.
The deletion of channel 3 has been done without considering the order in which it was inserted. One may verify that this is exactly the result that we would get if channels 1, 2 and 4 were added respectively, and channel 3 request had not come at all.

How does the scheme compare with the actual probabilities? One may verify that it is indeed exact when channels 1, 2 and 3 are added. The only source of error is channel 4, since its \( w \) of 0.3 is not an exact multiple of 0.2. The exact probability is 0.0006 while the bound is 0.0084. The high error need not discourage us, since a smaller value of \( \delta \) produces results that are fairly accurate.