Real-Time Scheduling with Quality of Service Constraints

Jay M. Hyman, *Student Member, IEEE*, Aurel A. Lazar, *Senior Member, IEEE*, and Giovanni Pacifici, *Member, IEEE*

Abstract—**Can the introduction of traffic classes improve upon the performance of ATM networks? We investigate this issue within the framework provided by a class of networks that guarantees quality of service. To provide a meaningful comparison we define the concept of *schedulable region*, a region in the space of loads for which the quality of service is guaranteed. We show the dependence of the schedulable region on the scheduling algorithm employed, quality of service parameters, and traffic statistics. An efficient real-time scheduling algorithm is introduced that substantially increases the schedulable region without incurring prohibitive complexity costs. The schedulable region associated with this algorithm is compared with the ones generated by the static priority scheduling algorithm and a variant of the minimum laxity threshold algorithm. The size and shape of the schedulable region is explored by means of simulations.**

I. Introduction

IN [1], a class of integrated networks was proposed for implementation with the capability of *efficiently* providing quality of service (QOS). The switching architecture of these networks is novel in that the concept of quality of service explicitly appears in the design specification at both the edge and core of the network. One of the fundamental requirements of these systems is that the *core* of the network makes a distinction between traffic classes. This was found necessary in order to efficiently provide quality of service. The immediate question these networks raise is whether traffic classes should also be introduced in ATM networks. This paper attempts to provide information for a better understanding and elucidation of this question. Our present study is limited to a switching node taken in isolation.

To date, there are two networks that were designed based on the ideas presented in [1]. One is MAGNET II [2], a fully instrumented and operational network testbed for metropolitan area applications. The other is TeraNet [3], a gigabit network currently under development. These networks are called asynchronous time sharing (ATS)-based because of the way the main network resources (switching and communication bandwidth, buffer space, and processing capacity) are allocated. The design of ATS-based networks heavily relies on the hardware implementation of buffer management and scheduling algorithms in which the QOS guarantee is explicitly incorporated. This represents its distinctive feature.

In order to give a quantitative framework for evaluating the performance of ATS-based networks that provide QOS guarantees, the concept of the *schedulable region* is introduced. It is the region in the space of possible loads for which a scheduling algorithm guarantees quality of service. The set of QOS constraints for the schedulable region is interpreted as a generalization of the classical "average time delay smaller than infinity" constraint that defines stability in the classical sense for queueing systems. The size and shape of the schedulable region depends, as shown in this paper, on the scheduling algorithm used, the QOS parameters under consideration, and the statistics of the traffic load. Throughout this paper, comparisons of different algorithms and evaluations of these various effects are expressed in terms of the size of the schedulable region. Presently, the exploration of the schedulable region is possible for the type of traffic sources considered in this paper only by means of simulations. Analytically tractable traffic sources for a FIFO queueing system are analyzed within the framework of Palm probabilities in [4].

We present the MAGNET II real-time scheduling algorithm (MARS), and compare its performance with that of other known algorithms that have already been considered in the literature, static priority scheduling (SPS) and a variant of minimum laxity threshold (MLT). Related work on scheduling was previously published in [5]–[9], [14], [15]. The MARS algorithm uses a simple knowledge structure, and was adopted for implementation on our real-time network testbed MAGNET II as well as TeraNet. The complexity of its implementation *versus* the corresponding performance is also investigated.

As already mentioned, this work is intended to provide data for the ongoing discussion about the need for introducing traffic classes in ATM networks. The introduction of traffic classes leads to higher complexity. Is this complexity warranted? The answer does not appear to be an easy one. From a strict performance point of view, we show that the schedulable region is increased and this in-

Manuscript received August 1990; revised May 1991. This paper was supported by the National Science Foundation under Grant ECD-88-11111. This paper was presented in part at the Seventh International Teletraffic Congress, Morristown, NJ, Oct. 9–11, 1990.

The authors are with the Center for Telecommunications Research, Columbia University, New York, NY 10027-6699.

IEEE Log Number 9102694.

crease leads to a substantial gain in the overall efficiency of utilization of network resources. More investigations along the lines described here will be needed, with data ultimately obtained from the operational networks that we have implemented.

This paper is organized as follows. In Section II, we briefly describe the main characteristics of the class of ATS-based broadband networks by using the block diagram implementation of the TeraNet switching node. The quality of service constraints, the link scheduling model, and the real-time traffic source models for benchmarking are also introduced. In Section III, the concept of the schedulable region is introduced, the SPS and MLT algorithms are described, and the MARS algorithm is introduced and presented in detail. In Section IV, the algorithms discussed in the previous section are evaluated *via* simulations. First, the complexity *versus* performance tradeoff for the MARS algorithm is discussed. Then, the dependence of the schedulable region on the scheduling algorithm, QOS parameters, and traffic statistics is evaluated. Finally, the distribution of the QOS parameters associated with each call is explored. Section V concludes the paper with a discussion of issues for further study.

II. The Generic Scheduling Problem

The generic scheduling problem presented in this section was motivated by the implementation of a class of asynchronous time sharing networks. The switching architecture of these networks is briefly described in Section II-A. Four traffic classes are introduced via quality of service constraints in Section II-B. Note that, in order to keep the complexity of the network manageable, the QOS for these classes is defined for the network as a whole, rather than for each individual call. The introduction of traffic classes calls for the introduction of scheduling algorithms. Models for scheduling of the link bandwidth are described in Section II-C. Source models of traffic flows for different traffic classes are discussed in Section II-D.

A. The Architecture of the Switching Node

The basic architeture of the ATS-based switching node was first implemented on MAGNET II [2]. It was also adopted for a new prototype multihop lightwave network called TeraNet [3]. In order to exemplify the application of the scheduling algorithms investigated in this paper, the architecture of the network interface units (switching nodes) of the latter network is briefly discussed here (see Fig. 1).

Each network interface unit consists of three input links, three output links, and a bus-based nonblocking switch fabric. Congestion may arise only at the output links. The architecture supports multiple traffic classes. Traffic arriving at an access point is transferred and stored, according to its class, into one (or more) of the multiple buffers. Each group of multiple buffers is connected to an output port. Note that this architecture allows for *simultaneous* arrivals at each of the output buffers during a packet transmission time. In Fig. 1, the critical resources (buffer space and communication links) are controlled by a link scheduler and buffer manager. Scheduling and buffer management policies have hardware support. A general discussion of the basic architecture of an ATS-based switching node is given in [1].

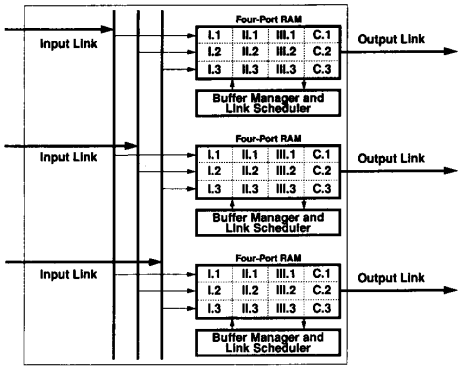


Fig. 1. The architecture of the TeraNet network interface unit.

B. The Quality of Service Constraints

The architecture shown in Fig. 1 supports four classes of traffic. Three of the traffic classes (Class I, II, and III) transport user traffic and are defined by a set of performance constraints. The fourth class, Class C, transports traffic of the network management system, and is not subject to specific QOS constraints.

Class I traffic is characterized by 0% *contention packet loss* and an end-to-end time delay distribution with a narrow support. The maximum end-to-end time delay between the source and destination stations is denoted by S^{I}. Class II traffic is characterized by ϵ% contention packet loss and an upper bound, η, on the average number of consecutively lost packets. It is also characterized by an end-to-end time delay distribution with a larger support than Class I. The maximum end-to-end time delay is S^{II}. Here, ϵ and η are arbitrarily small numbers and $S^{\mathrm{I}} \leq S^{\mathrm{II}}$. For Class I and II traffic, there is no retransmission policy for lost packets. Class III traffic is characterized by 0% *end-to-end packet loss* that is achieved with an end-to-end retransmission policy for error correction. If requested, it is also characterized by a *minimum* average user throughput Γ and a *maximum* average user time delay T.

C. The Link Scheduling Model

The contention problem at one of the output links of Fig. 1 is modeled as a resource allocation problem in a queueing system consisting of four FIFO buffers (one for each traffic class) sharing a single server as shown in Fig. 2. The server models a slotted channel with a fixed capacity of C bits/second and a fixed cell size of D bits/cell. The service rate is denoted by $\mu = C/D$ cells/second.

For this buffer model, there are various scheduling

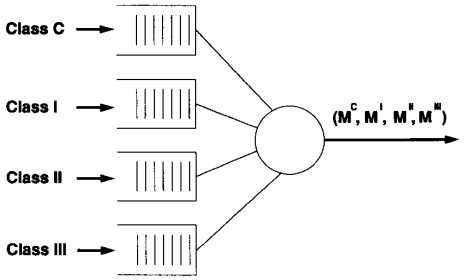


Fig. 2. The ATS link scheduling model.

mechanisms that can be employed. For some schedulers, such as SPS and MLT, control decisions are made at each packet departure time, and the parameters (M^C, M^I, M^{II}, M^{III}) are not relevant. For others, such as MARS, which utilize the asynchronous time sharing (ATS) scheme described below, control decisions occur at the end of each cycle, when these parameters are set for the course of the next cycle. In all cases, service is nonpreemptive; a cell transmission in progress cannot be aborted.

In the ATS scheme, the allocation of the server to each of the buffer (packet) classes is achieved by dividing time into periods called cycles consisting of up to H cells. Each cycle is further divided into four subcycles. During each subcycle (C, I, II, III), the link is allocated to the corresponding traffic class (C, I, II, III). For example, during subcycle I, Class I packets enter the server (link). The link scheduler uses four variables (M^C, M^I, M^{II}, M^{III}) to determine the boundary positions between subcycles. These variables represent the length (in cells) of subcycle C, I, II, and III, respectively, and can be dynamically adjusted by the scheduler according to the traffic load and mix. For a broader view of the switching and communication bandwidth allocation concepts included in the ATS framework, see [1].

D. Real-Time Traffic Source Models

For the purposes of this paper, we consider each of the traffic classes, defined via quality of service constraints in Section II-B, to carry information of a very specific type. Class I is assumed to consist of K^I video calls, and Class II of K^{II} voice calls. Class III consists of K^{III} data sources. The use of traffic source models raises two issues: how close are they to traffic sources encountered in practice, and what is the traffic mix employed.

The traffic models described below have been validated based on an extensive set of real-time measurements and qualitative evaluations of real-time video as well as traffic source models for video, voice, and data on MAGNET II [10]. Note, however, that the traffic sources used in the results presented in this paper display homogeneity. That is, a single source model is used within the same traffic class. Therefore, e.g., the same peak load values or interarrival time correlations are used. There is a simple reason for this deviation from what would most likely arise in practice: we could not find a more easily understandable way to interpret and present the results.

A single video call is modeled as a periodic random process that is characterized by a fixed frame duration of $F = 62.5$ ms, a constant bit rate of $c^I = 10$ Mb/s, and average active period $\mathbb{E}[\Sigma_{active}]$. At time $t = jF$, the frame begins and the source emits cells at a rate c^I/D. During this time, the source is active. The active period, Σ_{active}, is a random variable uniformly distributed between $\Sigma_{min} = 10$ ms and $\Sigma_{max} = 40$ ms. At the end of the active period, the source stops emitting cells for the duration of the frame, e.g., $F - \Sigma_{active}$. The cycle then repeats. The source average rate is given by:

$$\mathbb{E}\lambda(t, \text{video}) = \frac{\mathbb{E}[\Sigma_{active}]}{F} * c^I = 4 \text{ Mb/s}. \quad (2.1)$$

Multiplexing of K^I video calls is assumed to be accomplished by uniformly distributing the frame start at intervals of F/K^I ms.

A single voice call is modeled as an on–off source with constant arrivals. It is characterized by an exponentially distributed active period, in which cells are generated with constant rate c^{II}/D with $c^{II} = 64$ Kb/s, and an exponentially distributed silence period, in which no cells are generated.

Σ_{on} and Σ_{off} are two statistically independent random variables with a negative exponential distribution describing the on–off behavior of the source. $\mathbb{E}[\Sigma_{on}] = 352$ ms and $\mathbb{E}[\Sigma_{off}] = 650$ ms denote the average values of Σ_{on} and Σ_{off}, respectively [11]. The average rate for one such source is given by:

$$\mathbb{E}\lambda(t, \text{voice}) = \frac{\mathbb{E}[\Sigma_{on}]}{\mathbb{E}[\Sigma_{on}] + \mathbb{E}[\Sigma_{off}]} * c^{II} = 22.5 \text{ Kb/s}. \quad (2.2)$$

Finally, the data traffic is modeled as a Poisson source with average rate

$$\mathbb{E}\lambda(t, \text{data}) = 1 \text{ Mb/s}. \quad (2.3)$$

III. Scheduling Algorithms and the Schedulable Region

In this section, the main scheduling algorithms are introduced. The presentation in Section III-A starts with the introduction of the schedulable region concept that provides the framework for comparing the different scheduling algorithms. In Section III-B, two algorithms are introduced. One is based on static priority scheduling and the other on a variant of the minimum laxity threshold algorithm. In Section III-C, the MAGNET II real-time scheduling algorithm is presented and discussed in detail.

A. The Schedulable Region

Intuitively, the *schedulable region* of a queueing system is the set of points in the space of possible loads for

which the quality of service is guaranteed. As such, this concept is a generalization of the concept of stability region. Recall that the general concept of stability calls for finding the region in the space of loads for which the average time delay is finite. In our case, the set of constraints that determine the schedulable region is defined by the QOS constraints. Examples of constraints were given in Section II-B and include hard time delay constraints, probability of blocking and average gap constraints, average throughput, and average time delay constraints. Note that the schedulable region might be finite even for the case of a queueing system with finite buffer size. This is because the QOS constraints might restrict the loading on the system before the finite buffer size does.

Figs. 10, 13, and 14 are examples of schedulable regions for a three-dimensional queueing system with Class I, II, and III buffers. The axes in these figures show the load for each traffic class. The region in the three-dimensional space below the shaded surface represents the schedulable region. The size of the schedulable region depends on the scheduling algorithm used, the values of the QOS parameters, and the statistics of the traffic load. The dependency will be investigated in this paper. How to increase the schedulable region through the use of scheduling algorithms will be discussed in detail.

The theoretical characterization of the schedulable region appears, in general, to be fairly complex. A theoretical study of the schedulable region for the $MM/G/1$ queueing system can be found in [4]. Markov-modulated arrivals, hard-time delays, blocking, and average packet gap constraints were considered. In this paper, the shape and size of the schedulable region is explored with very few restrictions on the arrival process, QOS parameters, and class of scheduling algorithms. As a result, at this point in the theoretical development, only simulations can give an insight into the form and size of the schedulable region under these assumptions. The size of the schedulable region is a prime factor in determining the admission control policy for ATS-based networks [12].

For the sake of simplicity, we will limit our present study of the schedulable region to a queueing system with three user classes (Class I, II, and III). Therefore, we will assume that the system in Fig. 2 is not loaded with Class C traffic. This represents an approximation to the case where the Class C traffic load is negligible when compared with the aggregated user traffic load. An extension of our results to the general case of four classes is straightforward. Finally, we will assume that each of the buffers has infinite capacity. The finite buffer case, currently under consideration, will be published elsewhere.

B. SPS and MLT Algorithms

When static priority scheduling (SPS) is employed, Class I cells are always transmitted ahead of Class II, and Class II cells are always transmitted ahead of those of Class III. This scheduling scheme is simple to implement, and is thus often considered for scheduling of real-time traffic. Note that SPS scheduling is class-dependent. Intuitively, this policy will not be efficient when S^{I} is large compared to S^{II}. This is because the priority policy could cause QOS violations for the other traffic classes even while Class I delays are far from their allowed limits. In these cases, overall performance can be improved by delaying Class I packets within their QOS bounds.

The minimum laxity threshold (MLT) policy used in this paper will now be defined. Note that both the definition of laxity and the algorithm itself differ slightly from those used to define a previously published policy with the same name [6]. Let Q^k denote the number of cells in the Class k queue, and let $t^k(i)$ be the deadline of the ith Class k cell, for $i = 0, \cdots, Q^k - 1$. It is assumed that the cells in the buffer are sorted according to their deadlines. In our case, this is always true since the buffer behaves as a FIFO and the Class I maximum delay (S^{I}) is the same for every cell. Also, with no loss of generality (since service times are deterministic), we adopt the convention that deadlines are to the beginning of service.

The *laxity* $L^{\mathrm{I}}(i)$ of the ith Class I cell in the buffer at time t is defined by:

$$L^{\mathrm{I}}(i) = t^{\mathrm{I}}(i) - t - \frac{i}{\mu}. \tag{3.1}$$

Thus, for a given cell which knows that i other cells must precede it into service, the laxity represents the amount of time for which the server may remain idle, or serve cells of other classes, and still be able to serve this cell by its deadline. Note that this concern is not explicitly addressed by algorithms based only on the laxity of the packet at the head of the queue, as in [6].

The laxity for a Class II cell reflects the fact that it must be preceded into service by a number of Class I cells, in addition to all Class II cells ahead of it. Let $N^{\mathrm{I}}(t)$ denote the number of Class I cells which have deadlines before time t. The laxity for cell i in the Class II queue is then given by:

$$L^{\mathrm{II}}(i) = t^{\mathrm{II}}(i) - t - \frac{1}{\mu}(i + N^{\mathrm{I}}(t^{\mathrm{II}}(i))). \tag{3.2}$$

Prior to each cell transmission time, the laxities are evaluated for each of the cells in the queues for traffic classes I and II, and the minimum laxities L^{I} and L^{II} are computed for the queues themselves, by:

$$L^k = \min_{0 \le i < Q^k} L^k(i). \tag{3.3}$$

A threshold operation is then used to choose which class to serve. If $L^{\mathrm{I}} < 1/\mu$, then the Class I queue must be served. If not, and $L^{\mathrm{II}} < 1/\mu$, then the Class II queue must be served. If neither of these conditions are true, then a Class III cell may be transmitted (if one exists).

This scheduling policy is closely related to the OPT policy proposed in [9]. That policy, which is based on full knowledge of all future arrivals, is presented there as an

unrealizable optimal algorithm against which to benchmark other schedulers. Under the simplifying assumption made here that the cells of a given class always arrive in the order of their deadlines, and the assumption in [9] that all future arrivals are known (which, in this context, would be reflected by the exact knowledge of $N^{\mathrm{I}}(t)$ and the evaluation of laxities for packets not yet in the queues), the two policies should achieve identical schedules. However, while the OPT algorithm requires that the Class I schedule be completely filled in for all time before any Class II cells can be scheduled and is thus unrealizable, the MLT algorithm presented here allows for an implementation based on predicting $N^{\mathrm{I}}(t)$ when necessary. Specifically, if $t^{\mathrm{II}}(i) > t + S^{\mathrm{I}}$ (which is only possible if $S^{\mathrm{I}} < S^{\mathrm{II}}$), then $N^{\mathrm{I}}(t^{\mathrm{II}}(i))$ is given by the number of packets in the Class I queue plus the predicted number of Class I arrivals in the interval $(t, t^{\mathrm{II}}(i) - S^{\mathrm{I}}]$. The scheduler implemented for our studies evaluates laxities only for cells already in the buffers (3.3) and, when appropriate, uses an estimate of $N^{\mathrm{I}}(t)$ based on a first-order filter. This scheduler is thus expected to be very efficient, but estimation errors could decrease the link efficiency.

C. MARS: A Real-Time Scheduling Algorithm

In this section a real-time scheduling algorithm for ATS-based networks, called MARS, is presented. Referring to the ATS link scheduling model of Fig. 2, the server activity over the course of a cycle is divided into subcycles, whose lengths are defined by the parameters M^{I}, M^{II}, and M^{III}. The scheduler is responsible for properly setting these parameters.

Informally, the scheduler operates as follows. A maximum cycle length of H cells is first chosen. The knowledge structure available to the scheduler at the beginning of each cycle consists of two schedules (lists) of dimensions h^{I} and h^{II} that contain the number of Class I and Class II cells which arrived in each of the previous h^{I} and h^{II} cycles. The scheduling algorithm updates these lists at the end of each cycle by taking into account the number of new cells that arrived during the previous cycle. The scheduling algorithm is based on the intuition that in order to achieve high throughput, each cycle should serve only the Class I and II cells whose transmission cannot be further delayed. Thus, in each cycle, the scheduler first sets M^{I}, and then M^{II}; in each case choosing the minimum number of cells that must be transmitted to satisfy the QOS requirements. Any cells remaining in the cycle may then be assigned to M^{III}. Throughout this process, the scheduler must adhere to the maximum cycle length constraint

$$M^{\mathrm{I}} + M^{\mathrm{II}} + M^{\mathrm{III}} \leq H. \tag{3.4}$$

If, after M^{I} has been set, not enough cells remain in the cycle to satisfy the Class I requirements, the exceeding Class II cells are clipped.

The actual cycle length, often shorter than H, will change dynamically depending on the traffic load and profile. By keeping track of the number of packets arriving in each cycle, the scheduler will know ahead of time how many packets will be put up for service in the next cycle. To accomplish this, the scheduler updates the h^{I}- and h^{II}-dimensional schedules at the end of each cycle. These schedules correspond to a logical partitioning of the Class I and Class II buffers into *bins*.

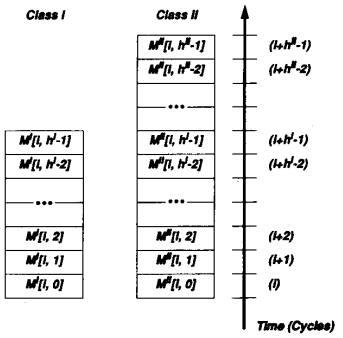


Fig. 3. Class I and Class II logical buffer partitions at the beginning of cycle i.

In the following, we will assume that $h^{\mathrm{II}} \geq h^{\mathrm{I}}$; the extension to the cases $h^{\mathrm{II}} = h^{\mathrm{I}}$ and $h^{\mathrm{II}} < h^{\mathrm{I}}$ is straightforward. Fig. 3 shows the Class I and Class II buffer logical partitions at the beginning of the ith cycle. The variable $M^k[i, j]$ represents the number of Class k cells that at the beginning of cycle i, $i \in N$, are *predicted* to be scheduled in the $(i + j)$th cycle, $j < h^k$, $k = \mathrm{I}, \mathrm{II}$. Thus, the jth bin contains the number of Class k cells that, according to the current buffer status, will be transmitted during the $(i + j)$th cycle. $M^k[i, j]$ is obtained through the *min* operator by comparing the value of $M^k[i - 1, j + 1]$ with the number of overflow packets created by the arrival of Class k cells, $A^k[i]$, during the $(i - 1)$th cycle. The updating algorithm for M^k, $k = \mathrm{I}, \mathrm{II}, \mathrm{III}$ is described below.

For Class I traffic, the updating process is described in Fig. 4. Since, as mentioned above, each bin contains the number of cells that are predicted to be scheduled for transmission for the next h^k cycles, each bin's maximum value cannot exceed the total server bandwidth available in each cycle (i.e., H). Reading the flow of the block diagram of Fig. 4 from left to right, we have the set of equations:

$$M^{\mathrm{I}}[i, j] = \min\left(M^{\mathrm{I}}[i - 1, j + 1] + O^{\mathrm{I}}[i, j + 1], H\right) \tag{3.5}$$

$$O^{\mathrm{I}}[i, j] = \max\left(0, M^{\mathrm{I}}[i - 1, j + 1] + O^{\mathrm{I}}[i, j + 1] - H\right) \tag{3.6}$$

for all j, $j = h^I - 1, h^I - 2, \cdots, 1, 0$ with $M^{\mathrm{I}}[i - 1, h^{\mathrm{I}}] = 0$ and $O^{\mathrm{I}}[i, h^I] = A^{\mathrm{I}}[i]$.

$O^{\mathrm{I}}[i, j]$ in (3.6) stands for overflow and represents the number of Class I cells that are predicted not to be transmitted during the $(i + j)$th cycle because this bin size would exceed the maximum bandwidth allocated to Class I for that cycle. It can be interpreted as the result of a

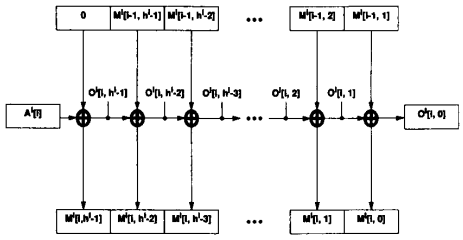


Fig. 4. Class I bin updating process at the beginning of cycle i.

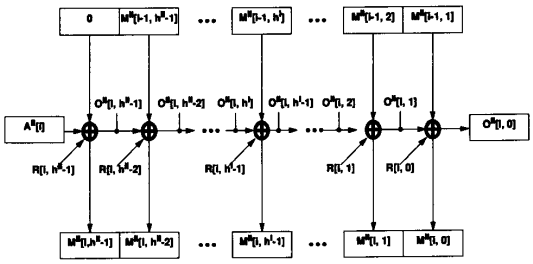


Fig. 5. Class II bin updating process at the beginning of cycle i.

ripple effect generated by a number of arrivals that is larger than the capacity (H) of the server. This ripple effect takes place when simultaneous arrivals occur due to bursts. As shown by (3.5), those cells will be scheduled for transmission in an earlier cycle m, where $i \le m < i + j$. Finally, M^{I}, the number of Class I cells scheduled for transmission during the ith cycle, is set to $M^{\mathrm{I}} = M^{\mathrm{I}}[i, 0]$.

Fig. 5 describes the Class II schedule updating process. This step is similar to the Class I bin updating process since, at this time, only the portion of server bandwidth not used by Class I traffic can be allocated to Class II. The Class II bins are updated as follows:

$$M^{\mathrm{II}}[i, j] = \min\left(M^{\mathrm{II}}[i-1, j+1] + O^{\mathrm{II}}[i, j+1], R[i, j]\right) \tag{3.7}$$

$$O^{\mathrm{II}}[i, j] = \max\left(0, M^{\mathrm{II}}[i-1, j+1] + O^{\mathrm{II}}[i, j+1] - R[i, j]\right) \tag{3.8}$$

for all j, $j = h^{\mathrm{II}} - 1, \cdots, 0$, with $M^{\mathrm{II}}[i-1, h^{\mathrm{II}}] = 0$ and $O^{\mathrm{II}}[i, h^{\mathrm{II}}] = A^{\mathrm{II}}[i]$. $R[i, j]$, the portion of server bandwidth predicted to be available to Class II in the $(i + j)$th cycle, is given by:

$$R[i, j] = \begin{cases} H - M^{\mathrm{I}}[i, j] & j = 0, 1, \cdots, h^{\mathrm{I}} - 1 \\ H - \dfrac{1}{h^{\mathrm{I}}} \displaystyle\sum_{l=0}^{h^{\mathrm{I}}-1} M^{\mathrm{I}}[i, l] & \\ & j = h^{\mathrm{I}}, \cdots, h^{\mathrm{II}} - 1. \end{cases} \tag{3.9}$$

Note that, if $h^{\mathrm{II}} > h^{\mathrm{I}}$, $R[i, j]$ for $j = h^{\mathrm{I}}, h^{\mathrm{I}} + 1, \cdots, h^{\mathrm{II}} - 1$ will depend on the number of future Class I arrivals. As this information is not yet available, it is estimated using a simple first-order estimator. The number of Class II cells scheduled for transmission during the ith cycle is set to $M^{\mathrm{II}} = M^{\mathrm{II}}[i, 0]$.

Finally, M^{III}, the number of Class III packets to be served in the current cycle, is set to

$$M^{\mathrm{III}} = \min\left(Q^{\mathrm{III}}[i], H - M^{\mathrm{I}} - M^{\mathrm{II}}\right) \tag{3.10}$$

where $Q^{\mathrm{III}}[i]$ is the queue size of the Class III buffer at the beginning of cycle i.

In what follows, the dimensioning of the number of bins that guarantees quality of service for Class I and II traffic will be discussed. As mentioned before, since the Class I cells arriving during the $(i - 1)$th cycle are transmitted no later than during the $(i + h^{\mathrm{I}} - 1)$th cycle, it follows that the worst-case delay corresponds to the case of a cell arriving at the beginning of the $(i - 1)$th cycle and being transmitted at the end of the $(i + h^{\mathrm{I}} - 1)$th cycle. In this case, the maximum Class I delay is given by

$$W^{\mathrm{I}}_{\max} = (h^{\mathrm{I}} + 1)\frac{H}{\mu} \tag{3.11}$$

provided that the average Class I load over h^{I} consecutive cycles does not exceed μ [see also (3.16)]. Therefore, under mild regularity assumptions on the traffic profile, the QOS for Class I traffic is guaranteed if:

$$(h^{\mathrm{I}} + 1)\frac{H}{\mu} \le S^{\mathrm{I}}. \tag{3.12}$$

The h^{I} value that satisfies (3.12) and gives the largest $W^{\mathrm{I}}_{\max}$ is given by:

$$h^{\mathrm{I}} = \left\lfloor \frac{S^{\mathrm{I}}}{H}\mu \right\rfloor - 1 \tag{3.13}$$

where $\lfloor x \rfloor$ denotes the largest integer less than or equal to z. From (3.11) and (3.13), it follows that the Class I maximum delay $W^{\mathrm{I}}_{\max}$ is a function of the maximum cycle length H:

$$W^{\mathrm{I}}_{\max} = \left\lfloor \frac{S^{\mathrm{I}}}{H}\mu \right\rfloor \frac{H}{\mu}. \tag{3.14}$$

Since $h^{\mathrm{I}} \ge 1$, it follows from (3.12) that the cycle length H is bounded by:

$$H \le \frac{S^{\mathrm{I}}}{2}\mu. \tag{3.15}$$

In our discussion up to this point, we have not addressed the properties of the overflow bin $O^{\mathrm{I}}[i, 0]$. As mentioned before, $O^{\mathrm{I}}[i, j]$ represents the number of Class I cells that cannot be transmitted during the $(i + j)$th cycle. Since the Class I bin values are bounded by H (i.e., $M^{\mathrm{I}}[i - 1, j + 1] \le H$), it follows from (3.5) and (3.6) that

$$O^{\mathrm{I}}[i, j] \ge 0 \Rightarrow O^{\mathrm{I}}[i, j+1] > 0 \tag{3.16}$$

and, therefore, if $O^{\mathrm{I}}[i, 0] > 0$

$$M^{\mathrm{I}}[i, j] = H \tag{3.17}$$

for all j, $0 \le j \le h^{\mathrm{I}} - 1$. Hence, $O^{\mathrm{I}}[i, 0]$ is greater than zero, and $O^{\mathrm{I}}[i, 0]$ Class I cells are going to experience a delay greater than S^{I}, only when the Class I traffic overloads the server (arrivals exceed the server capacity) for a period of time larger than $h^{\mathrm{I}}H/\mu$. Any traffic load allowing this condition to occur will be outside the schedulable region, and must be avoided by the use of an admission control algorithm that controls the traffic load [12].

Using the same reasoning as for (3.14), the maximum Class II delay $W^{\mathrm{II}}_{\max}$ is given by:

$$W^{\mathrm{II}}_{\max} = \left\lfloor \frac{S^{\mathrm{II}}}{H}\mu \right\rfloor \frac{H}{\mu} \tag{3.18}$$

provided that packets to be clipped are dropped from the Class II buffer.

Similarly, it can be shown by arguments parallel to (3.16) and (3.17) that the quantity $O^{\mathrm{II}}[i, 0]$ represents the number of Class II cells that would experience a delay in the worst case greater than S^{II}. Thus, the scheduler will drop this number of cells from the Class II buffer.

IV. Experimental Evaluation of the Scheduling Algorithms

In this section, we will present the results of simulation experiments which illustrate several properties of the scheduling algorithms and associated schedulable regions. As stated in Section III-A, the schedulable region for a given system is determined by three factors: the QOS requirements that the system must meet, the statistical characteristics of the offered traffic, and the choice of the scheduling algorithm. These dependencies are systematically addressed below.

In Section IV-A, the performance of the MARS algorithm is explored for different values of the maximum cycle length H. The choice of the operating value to be chosen for this parameter is shown to involve a tradeoff between performance and complexity. In Sections IV-B and IV-C, the schedulable regions for the three scheduling algorithms are presented for different sets of QOS constraints and traffic characteristics. When the relative performance of the three algorithms is assessed under these different conditions, it is found that while performance is always better for the more complex algorithms, this difference is most pronounced when the traffic is more bursty and the QOS constraints are relatively loose. In Section IV-D, we investigate the QOS provided to each individual user (or virtual circuit) when the class-based QOS constraints are satisfied. The three scheduling algorithms are compared in their ability to spread any performance degradations fairly over the user population. Once again, an advantage is found for the more complex algorithms.

All of the simulations reported in this section are based on the assumptions of a fixed cell size of $D = 1024$ bits/cell and a link capacity of $C = 100$ Mb/s. In addition, for every simulation run, a transient period of 2 s was allowed before any measurements were taken, and the simulations were run for 60 s. The 95% confidence bounds for the measured performance criteria were well within 5% of the observed values.

A. Complexity Versus Performance Tradeoff

In this section, the performance of the MARS scheduling algorithm is studied *versus* its complexity. The complexity of MARS is a function of the number of Class I and Class II bins. The higher the number of bins, the higher the number of operations that the scheduler has to perform at the beginning of each cycle. On the other hand, as shown in Section III-C, the number of bins is inversely proportional to the maximum cycle length H, as determined by worst-case delay considerations (3.13). This worst-case delay represents the pessimistic assumption that a Class I packet arriving at the beginning of the $(i - 1)$th cycle is not transmitted until the end of the $(i + h^{I} - 1)$th cycle. As Class I cells are always served at the beginning of a cycle, this event would require that H Class I cells arrive at the very beginning of cycle $i - 1$, and that this will occur at the tail end of a burst, with $(h^{\mathrm{I}} - 1)H$ Class I cells already in the buffer. This is, thus, a rather rare event. As the maximum cycle length increases, the probability that such worst-case delays occur decreases, and the estimation error due to this assumption, on the order of H/μ, increases. For Class II packets, that are served later in the cycle, the estimated worst-case delays are more accurate.

This behavior is exemplified in Fig. 6. Here, observed values of the Class I and II maximum delays and Class III average delay are plotted against the maximum cycle length H. In this experiment, Class I traffic was composed of $K^{\mathrm{I}} = 13$ video calls, while Class II load resulted from multiplexing $K^{\mathrm{II}} = 900$ voice calls. $K^{\mathrm{III}} = 20$ Poisson sources with aggregate average load of $\mathbb{E}\lambda(t, \text{data}) = 20$ Mb/s was mapped into Class III traffic. The QOS vector chosen for these experiments was $[S^{\mathrm{I}}, S^{\mathrm{II}}, \epsilon, \eta, T] = [1$ ms, 1 ms, 0.001, 5.0, 1 ms]. Fig. 6 shows that as H approaches one, the Class I and II maximum delays approach S^{I} and S^{II}, allowing a smaller average delay for Class III. As H increases, the scheduler does not delay Class I and II cells up to the QOS limit, resulting in a higher Class III average delay. Also, Fig. 6 shows that when the Class I and II maximum delays have a peak [when H is close to an integral multiple of $S^k\mu$ in (3.14) and (3.18)], the Class III average delay has a trough and vice versa. Furthermore, the sawtooth behavior of the Class I and II maximum delay can be explained by the fact that as the number of bins is kept constant and H is increased, the maximum delay linearly increases with H. As H becomes large enough, the scheduler will use one less bin (3.13), decreasing the maximum delay by a quantity close to H/μ.

Figs. 7 and 8 show the Class II clipping probability and average gap length as a function of H. It is evident that the Class II QOS degrades as H increases. This is due to

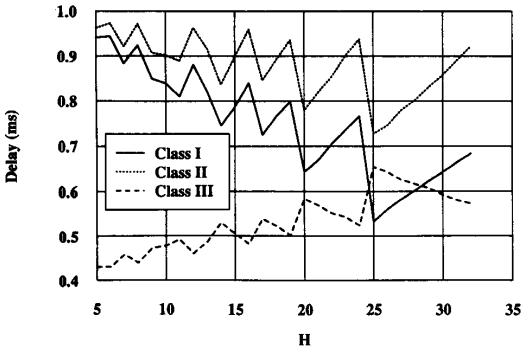


Fig. 6. Class I, II, and III delay *versus* H (K^{I} = 13, K^{II} = 900, K^{III} = 21, S^{I} = 1 ms, S^{II} = 1 ms).

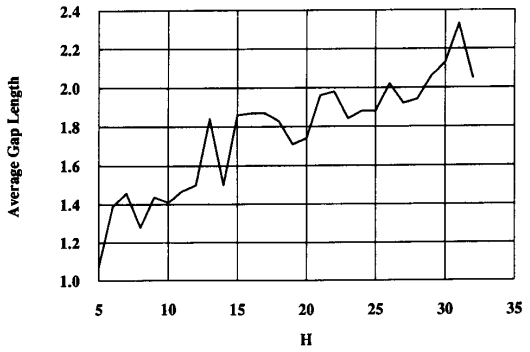


Fig. 8. Class II average gap length *versus* H (K^{I} = 13, K^{II} = 900, K^{III} = 21, S^{I} = 1 ms, S^{II} = 1 ms).

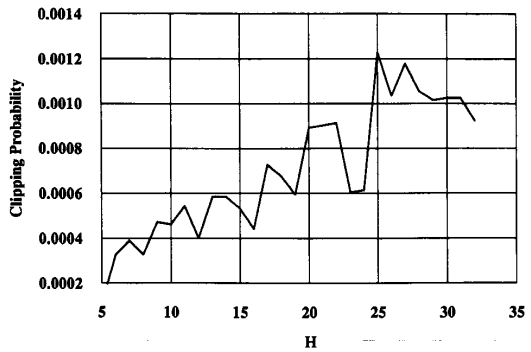


Fig. 7. Class II clipping probability *versus* H (K^{I} = 13, K^{II} = 900, K^{III} = 21, S^{I} = 1 ms, S^{II} = 1 ms).

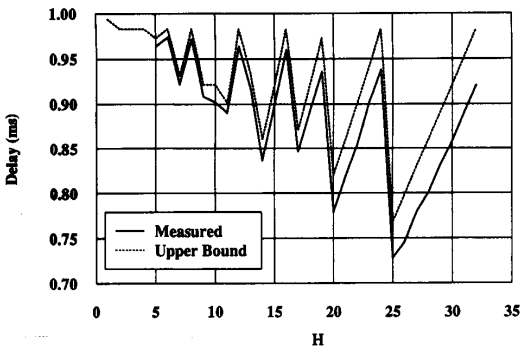


Fig. 9. Class II maximum delay *versus* H (K^{I} = 13, K^{II} = 900, K^{III} = 21, S^{I} = 1 ms, S^{II} = 1 ms).

two main reasons. First, as H increases, the scheduler delays Class I cells less, allowing less flexibility in serving Class II cells. Also, the scheduler drops, in each cycle, $O^{II}[i, 0]$ cells; this represents the number of Class II cells that would experience, in the worst case, a delay greater than S^{II}. As mentioned before, the probability that such a worst case occurs decreases as H increases. Thus, the scheduler might drop cells whose delay is not bigger than S^{II}.

Finally, Fig. 9 shows the estimated worst-case Class II maximum delay, given by (3.18), and the observed maximum delay for class II as a function of H. These results confirm the previous discussion. The Class II maximum delay is bounded by (3.18), but the error increases as H increases. Furthermore, the resulting error is always of the order of H/μ.

We repeated this experiment for different mixing of traffic sources. The results obtained confirmed the qualitative observation made for this experiment and they also showed a quantitative dependency on the source models and QOS parameters.

B. The Dependence of the Schedulable Region on the QOS Parameters

In this section, the schedulable regions are plotted for each of the control algorithms, using the traffic source models presented in Section II-D, for two different QOS vectors. To obtain each point in these plots, the values of K^{I} and K^{III} (the number of calls of Classes I and III) were fixed, and simulations were run to determine the maximum number K^{II} of voice calls for which the QOS could be satisfied over a period of time of 60 s. For each control algorithm and QOS vector, K^{I} was varied from 0 to 20 video calls, and K^{III} was varied from 0 to 80 Poisson source, to yield a three-dimensional depiction of the schedulable region. Unless otherwise stated, the parameter H of the MARS algorithm was set equal to 39.

The three-dimensional region obtained for the MARS algorithm with a QOS vector of $[S^{I}, S^{II}, \epsilon, \eta, T]$ = [2 ms, 4 ms, 0.001, 5.0, 8 ms] is shown in Fig. 10. The three-dimensional regions for the other two schedulers for the same QOS vector are somewhat similar in appearance and are not shown. Rather, to facilitate comparison of the schedulable regions for the different algorithms, the two-dimensional projections of these three surfaces onto the plane K^{III} = 0 are shown in Fig. 11. When K^{I}, the number of Class I calls in the system, is less than 10, the performance of these three scheduling algorithms is approximately the same. Under these conditions, the SPS scheduling algorithm allows a very high degree of network utilization ($\simeq$ 98%), leaving no room for further improvement. However, for larger values of K^{I}, as shown,

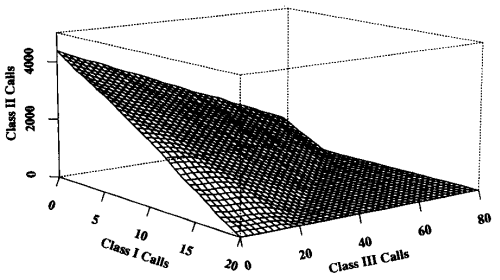


Fig. 10. Schedulable region for MARS, with QOS = [2, 4, 0.001, 5, 8 ms].

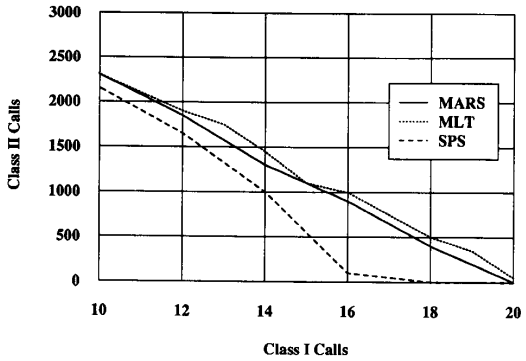


Fig. 11. Schedulable regions for $K^{III} = 0$, QOS = [2, 4, 0.001, 5, 8 ms].

MLT and MARS scheduling allow the QOS requirements to be satisfied with more Class II users in the network than could be allowed using SPS. For $K^{I} = 16$, for example, this performance gain corresponds to about 800 voice calls.

Fig. 12 shows a similar two-dimensional cross section of the schedulable regions for the more stringent QOS vector $[S^{I}, S^{II}, \epsilon, \eta, T] = [400\ \mu s, 800\ \mu s, 0.001, 5.0, 1.6\ ms]$. (To keep the number of bins constant, the MARS parameter H was set to 9 for this experiment.) In this case, the schedulable regions for the three schedulers are approximately the same. A comparison of this plot with the previous one reveals that the drastic decrease in the delay bounds S^{I} and S^{II} has little effect on the schedulable region for SPS, but greatly reduces the regions for MARS and MLT (thus reducing the throughput gain due to these schedulers from 800 voice calls down to about 200 calls).

The conclusion based on these experiments is that when Class I and II require small delays, the static priority scheduling policy is very nearly optimal. When the allowable delays for Class I and II traffic are somewhat larger, however, a significant gain in utilization can be achieved by using one of the more complex algorithms.

C. The Dependence of the Schedulable Region on the Traffic Characteristics

In this section, we use different parameters for the Class I traffic source model of Section II-D to illustrate the impact of different traffic source characteristics on the schedulable region. The Class I source parameters used in this section are motivated by observations in [10] that substantial degradation of network performance can be caused by the intersource correlation and burstiness of real-time video sources. In this section, the Class I source parameters are: $\Sigma_{min} = 2$ ms, $\Sigma_{max} = 8$ ms, $c^{I} = 50$ Mb/s, and $F = 62.5$ ms. Although the Class I source average rate is the same as in Section II-D ($\mathbb{E}\lambda(t$, video) = 4 Mb/s), the source active rate is increased and the average active period is correspondingly reduced to $\mathbb{E}[\Sigma_{active}] = 5$ ms. Finally, for Class II and Class III, we used the same source parameters as in Section II-D.

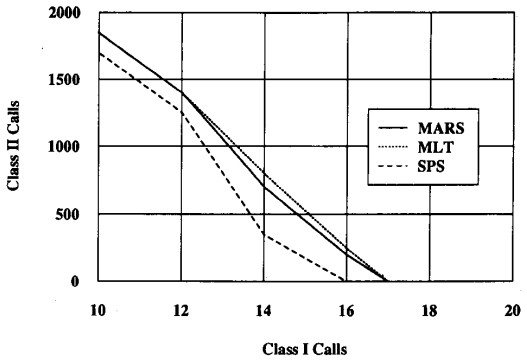


Fig. 12. Schedulable regions for $K^{III} = 0$, QOS = [400 μs, 800 μs, 0.001, 5, 1.6 ms].

The schedulable regions were plotted using this traffic source model for the QOS vector $[S^{I}, S^{II}, \epsilon, \eta, T] = [1$ ms, 1 ms, 0.001, 5.0, 1 ms]. Figs. 13 and 14 show the schedulable regions obtained for SPS and MLT, respectively. The MARS region, not shown, is almost identical to that obtained for MLT. The MLT throughput gain over SPS is evident over the entire K^{I}, K^{II}, and K^{III} range. Only when both Class I and Class II loads are low is the performance of SPS, MARS, and MLT the same. The comparison of the two-dimensional projections for $K^{III} = 0$ in Fig. 15 shows that, under certain loading conditions, the gain can be as high as 40% of the total bandwidth. This experiment suggests that, *when the network is loaded with highly correlated and bursty traffic, an information-based scheduling policy can greatly improve network utilization.* This conclusion is supported by Fig. 11 as well. Only when most of the load was provided by Class I traffic did MARS and MLT improve the system performance. When voice calls are multiplexed together, their correlation decreases as the number of sources increase, and the aggregate traffic is less bursty.

D. User-Oriented Quality of Service

The quality of service requirements presented in Section II-B, that were used to define the schedulable regions for each scheduler, reflect the performance of the network as seen by each *class* of traffic. In this section, we briefly investigate system performance as seen by each individ-

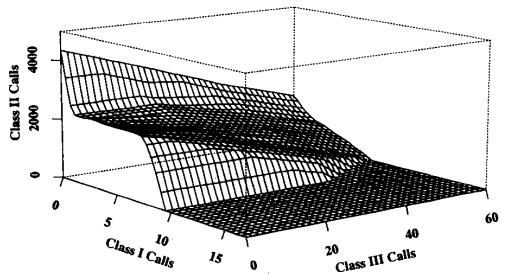


Fig. 13. Schedulable region for SPS, with highly correlated sources.

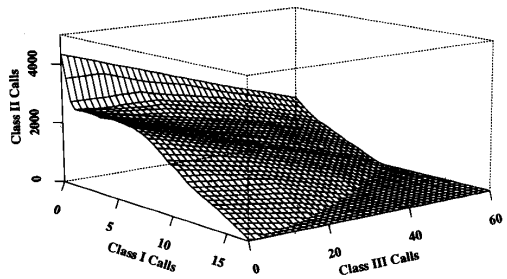


Fig. 14. Schedulable region for MLT, with highly correlated sources.

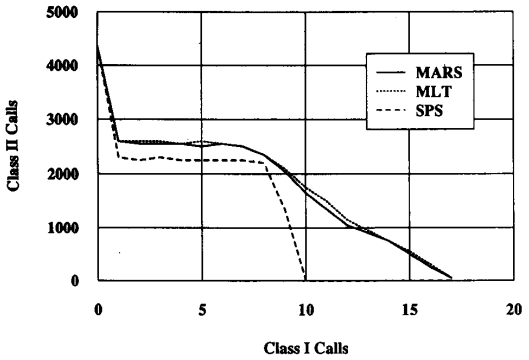


Fig. 15. Schedulable regions for $K^{III} = 0$, highly correlated traffic sources, QOS = [1, 1, 0.001, 5, 1 ms].

ual *call* or virtual circuit (VC) within a class. Two distinct issues are discussed. First, what does the quality of service guarantee for the class imply for each individual VC? Second, when service degradations occur, how fairly are they distributed among the various VC's?

The relationship between per-class and per-VC performance varies with the type of performance measure considered. Performance guarantees based on absolute bounds will hold for each VC as well as for the class as a whole. For instance, when the maximum delay bound S^I is met for Class I, then this bound will clearly be met for each VC as well. Thus, in this case, the class-based performance guarantee suffices to guarantee the same QOS to each individual VC.

For performance measures based on cell loss rates, this is no longer the case. The cell loss rate for any single call may be somewhat higher or lower than that of the class as a whole. Nevertheless, the cell loss rate seen by the class is a (weighted) average of the loss rates of all the VC's in the class. It should, thus, be possible to choose a bound on the class-based cell loss rate which can ensure that the cell loss rate for any given VC is less than ϵ with a given confidence level, say 95%.

For measures of consecutive cell loss, the relationship is even weaker. Consecutive cells in the aggregate traffic stream for a class will not generally belong to the same VC, and vice versa. The average gap length for Class II measures, in general, a different phenomenon than the average gap length as seen by each VC. Thus, unlike the case for cell loss rates, the class-based measurements do not represent an average of the per-VC measurements. Nevertheless, it is instructive to observe how the average gap length varies from one VC to another when the class-based QOS requirements are met.

To give insight into the per-VC distributions of both the clipping probability (cell loss rate) and average gap length for Class II traffic, these quantities were measured separately for each multiplexed voice call during simulations using each of the control algorithms under investigation. Ideally, it would be desirable that the distributions for these performance measures be closely concentrated around their means, representing a fair sharing of any performance degradations among the user population and ensuring that all calls of a class receive roughly equivalent quality of service. In practice, none of the control algorithms investigated achieves this ideal, but an algorithm which yields smaller tails in this distribution is preferred over one with longer tails.

For each scheduling algorithm, the system was loaded at a point near the boundary of the schedulable region for the QOS vector [2 ms, 4 ms, 0.01, 5.0, 8 ms]. To achieve this, K^I was fixed at 10 video calls, K^{III} was set to 0, and the number of voice calls K^{II} was chosen such that the aggregate average cell loss rate for Class II was almost exactly equal to the allowable loss rate of 0.01. The results are shown in Figs. 16 and 17. Both graphs show that the best performance, as indicated by the smallest tails in the distributions, is achieved by the MLT algorithm, followed by MARS, and then by SPS with the largest tails. Once again, the achieved performance is found to be commensurate with the complexity of the control algorithm.

We have seen that class-based QOS guarantees do not always translate directly into per-VC guarantees, but that for some performance measures the two are strongly linked. In addition, we have seen that good scheduling algorithms, even when based on per-class performance measures, can help avoid an unfair distribution of performance among the calls within a class. Nevertheless, for a full characterization of the user-oriented QOS which emerges from controls based on class-oriented QOS, a much more thorough investigation is necessary. In ongoing research, we are exploring the relationship between the QOS at these two levels, and how it is affected by the loading conditions and QOS parameters. Of course, it

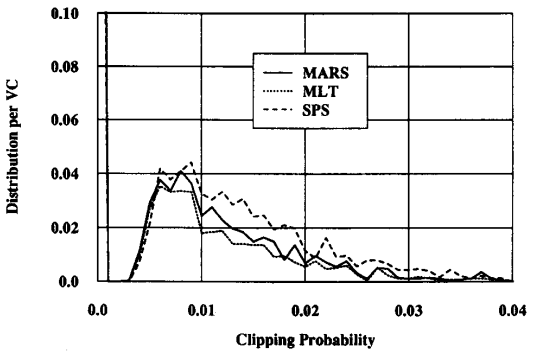


Fig. 16. Class II clipping probability distribution per virtual circuit.

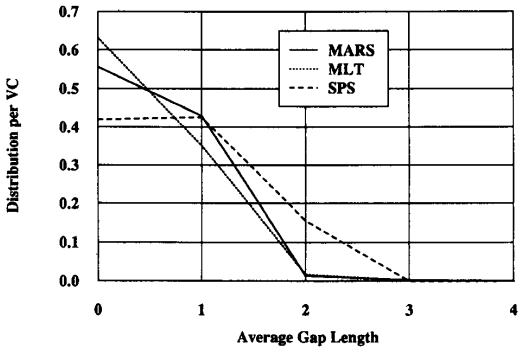


Fig. 17. Class II average gap length distribution per virtual circuit.

would be desirable to truly guarantee QOS on a per-call basis. However, this would require a substantial further increase in controller complexity.

V. Conclusions

Several issues raised in this paper merit further investigation, and will be investigated in the future. A brief discussion is provided here.

The concept of schedulable region appears to be a very powerful tool for investigating the "capacity region" of a link or switch. The schedulable region can be used for admission control. By estimating its size, the admission controller can decide whether a call should be accepted or rejected. We have seen that the schedulable region depends on the scheduling algorithm employed, the QOS parameters, and the traffic statistics. This observation has many practical consequences.

First, because of the many parameters that influence it, a universal approximation of the schedulable region does not appear to be easy to extract. The schedulable region has been approximated by a hyperplane in [13]. The results obtained here suggest that under certain traffic loads and profile and QOS constraints, this approximation might lead to a substantial under- or over-utilization of resources. Work is under way to understand whether better approximations are possible and/or desirable. Second, as the traffic mix can have a major impact on the schedulable region, the latter can only be estimated as well as one can estimate the traffic mix. Thus, in practice, it is advisable to specify an operating region which allows some margin around the boundary of the schedulable region, and an admission control policy which restricts the network load to within the operating region. Third, *adaptive scheduling algorithms* of the type proposed here are called for in networks with unknown traffic statistics. This will guarantee a large schedulable region under many different operating conditions.

What the structural results that appear to emerge from our study on the dependence of the schedulable region on the scheduling algorithms? Recall that the SPS algorithm always allocates resources to Class I traffic first, while the MARS and MLT algorithms allocate to Class I traffic only the amount of resources that are necessary to satisfy the Class I QOS parameters. The simulation results show that when SPS is used, Class I traffic always experiences a very small maximum delay ($< 100\ \mu s$), while with MARS the Class I maximum delay approaches S^I. This allows Classes II and III to have more resources allowed to them, and thus the multiplexer has a greater link utilization factor. Finally, MLT scheduling achieves the largest schedulable region among the studied algorithms.

MLT is a very complex algorithm, since it involves maintaining the laxities of each queued Class I and Class II cell. The number of these operations, which must be computed prior to each cell transmission, is equal to the number of queued packets, which in any real implementation will be bounded by the buffer capacity. In contrast, MARS allocates network resources only at the end of each cycle, and the number of bins in the information structure it maintains is generally smaller than the buffer size. The algorithm therefore requires less computation, less often, than does MLT. It was no surprise, therefore, that the simulations using the MARS scheduler required, at most, twice the computation time of the SPS scheduler while the MLT runs took substantially longer. Theoretical studies will be needed, however, to make these observations more precise.

The MLT scheduler can be seen as achieving a close to upper bound network utilization. Our results show that the MARS scheduler is also close to this upper bound. Considering that the additional knowledge requirement to implement the MLT algorithm leads to a substantial increase in complexity without a proportional improvement in network utilization, we recommend the MARS scheduling algorithm for implementation in ATS-based switching nodes.

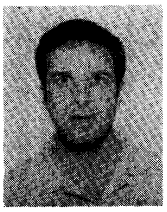
Acknowledgment

The authors would like to thank the reviewers for their helpful comments.

References

[1] A. A. Lazar, A. T. Temple, and R. Gidron, "An architecture for integrated networks that guarantees quality of service," *Int. J. Digit. Analog Commun. Syst.*, vol. 3, pp. 229-238, April-June 1990.

[2] ——, "MAGNET II: A metropolitan area network based on asynchronous time sharing," *IEEE J. Select. Areas Commun.*, vol. 8, pp. 1582–1594, Oct. 1990.
[3] R. Gidron and A. T. Temple, "TeraNet: A multihop multichannel lightwave network," in *Proc. IEEE Int. Conf. Commun.*, Denver, CO, June 1991, pp. 602–608.
[4] J. M. Ferrandiz and A. A. Lazar, "Admission control of real-time sessions of an integrated node," in *Proc. IEEE INFOCOM '91*, Bal Harbour, FL, Apr. 7–11, 1991, pp. 553–559.
[5] A. A. Lazar, A. Patir, T. Takahashi, and M. E. Zarki, "MAGNET: Columbia's integrated network testbed," *IEEE J. Select. Areas Commun.*, vol. SAC-3, pp. 859–871. Oct. 1985.
[6] R. Chipalkatti, J. F. Kurose, and D. Towsley, "Scheduling policies for real-time and non- real-time traffic in a statistical multiplexer," in *Proc. IEEE INFOCOM '89*, Ont., Canada, Apr. 1989, pp. 774–783.
[7] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE J. Select. Areas Commun.*, vol. 8, pp. 368–379, Apr. 1990.
[8] K. Sriram, "Dynamic bandwidth allocation and congestion control scheme for voice and data multiplexing in wideband packet technology," presented at the IEEE Int. Conf. Commun., Atlanta, GA, Apr. 1990.
[9] F. A. Tobagi and J. M. Peha, "Evaluating scheduling algorithms for traffic with heterogeneous performance objectives," in *Proc. GLOBECOM '90*, San Diego, CA, Dec. 1990, pp. 21–27.
[10] A. A. Lazar, G. Pacifici, and J. S. White, "Real-time traffic measurements on MAGNET II," *IEEE J. Select. Areas Commun.*, vol. 8, pp. 467–483, Apr. 1990.
[11] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data," *IEEE J. Select. Areas Commun.*, vol. SAC-4, pp. 833–846, Sept. 1986.
[12] J. M. Hyman, A. A. Lazar, and G. Pacifici, "Joint scheduling and admission control for ATS-based switching nodes," to appear in *IEEE J. Select. Areas Commun.*
[13] C. Courcoubetis, G. Fouskas, A. A. Lazar, S. Leventis, and S. Sartzetakis, "WIENER and NEMESYS: A comparison of two quality-of-service network management experiments," presented at the Fourth RACE Telecommun. Manag. Network Conf., Dublin, Ireland, Nov. 14–16, 1990.
[14] S. Golestani, "A stop-and-go queueing framework for congestion management," in *Proc. SIGCOMM'90*, Philadelphia, PA, Sept. 1990, pp. 8–18.
[15] L. Zang, "Virtual Clock: A new traffic control algorithm for packet switching networks," in *Proc. SIGCOMM'90*, Philadelphia, PA, Sept. 1990, pp. 19–29.

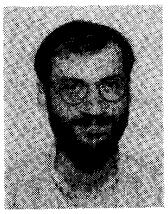


Jay M. Hyman (S'82) was born in New York, NY, on June 26, 1961. He received the B.S., M.S., and M.Phil. degrees from the Department of Electrical Engineering, Columbia University, New York, in 1983, 1984, and 1987, respectively.

He is currently nearing completion of the Ph.D. degree at Columbia University. He spent the summer of 1984 at IBM working with a team on adaptive distributed routing. In the summer of 1985, he worked at AT&T Bell Laboratories in the area of protocol testing and validation. As a Graduate Resaerch Assistant in the Center for Telecommunications Research at Columbia University, he participated in research projects on speech processing and neural networks. His current research interests are in the areas of scheduling and admission control for integrated telecommunication networks.

Mr. Hyman is a member of Tau Beta Pi and Eta Kappa Nu.

Aurel A. Lazar (S'77–M'80–SM'90), for a photograph and biography, see this issue, p. 967.



Giovanni Pacifici (S'80–M'85) was born in Rome, Italy, on Sept. 27, 1957. He received the Dr.Eng. and Ph.D. degrees from the Department of Information and Communication Technology, University of Rome La Sapienza, in 1984 and 1989, respectively.

As a student, his main activities focused on the performance evaluation of local and metropolitan area networks, with an emphasis on the integration of voice and data. During his course of studies, he was a Visiting Scholar at the Center for Telecommunications Research, Columbia University, New York, from 1987 to 1988. In 1989, he joined the staff of the Center for Telecommunications Research as a Research Scientist. His current interests include monitoring and control of broadband integrated networks, real-time traffic generation, parallel processing, and switching architectures.