A Near-Optimal Packet Scheduler for QoS Networks*

Dallas E. Wrege *Jörg Liebeherr*

Department of Computer Science
University of Virginia
Charlottesville, VA 22903
Email: {dallas | jorg}@cs.virginia.edu

Abstract

A packet scheduler in a quality-of-service (QoS) network should be sophisticated enough to support stringent QoS constraints at high loads, but it must also have a simple implementation so that packets can be processed at the speed of the transmission link. The *Earliest-Deadline-First* (EDF) scheduler is the optimal scheduler for bounded-delay services in the sense that it provides the tightest delay guarantees of any scheduler, but an implementation of EDF requires the sorting of packets, a complex operation that is not practical for high-speed networks. In this study we present the design, implementation, and analysis of the novel *Rotating-Priority-Queues*$^+$ (RPQ$^+$) scheduler that is near-optimal in the sense that it can approximate EDF with arbitrary precision. The RPQ$^+$ scheduler uses a set of prioritized FIFO queues whose priorities are rearranged (*rotated*) periodically to increase the priority of waiting packets. We show that RPQ$^+$ has the following desirable properties: its implementation requires operations independent of the number of queued packets, it can provide worst-case delay guarantees, and it is always superior to a *Static-Priority* (SP) scheduler. For shared-memory architectures, we show that RPQ$^+$ can be implemented with little computational overhead. We derive expressions for the worst-case delays in an RPQ$^+$ scheduler and demonstrate that the achievable network utilization increases with the frequency of queue rotations, approaching that of EDF in the limit. We use numerical examples, including examples based on MPEG video, to show that in realistic scenarios RPQ$^+$ can closely approximate EDF even for infrequent queue rotations.

Key Words: Multimedia Networks, Scheduling, Multiplexing, Bounded-Delay Service, Quality-of-Service, Deterministic Service.

*This work is supported in part by the National Science Foundation under Grant No. NCR-9309224.

1 Introduction

Future integrated services networks based on Asynchronous Transfer Mode (ATM) technology are expected to support applications with a wide range of service requirements. The class of multimedia applications such as voice and video is particularly demanding since these applications require guarantees on their *quality-of-service* (QoS) in terms of bounded delay, loss, and jitter. To provide QoS, a network must reserve resources such as bandwidth and buffer space for individual connections; however, since resource availability is limited, the network must carefully manage its resources in order to ensure high achievable network utilization. One of the most important components in the design of such a network is the choice of *packet schedulers* at network switches that determine the transmission order of packets queued at output buffers. In this study, we consider the design of packet schedulers appropriate for use in networks with a *bounded-delay service*, that is, networks that provide deterministic (i.e., worst-case) delay guarantees to all packets on a connection.

Many packet schedulers have been considered for use in bounded-delay services [7, 8, 13, 15, 29]. The well-known *Earliest-Deadline-First* (EDF) scheduler has been studied in [7, 11, 12, 18] and is distinguished in that it has optimal *efficiency*: for a given set of connections, EDF can support delay guarantees that are at least as tight as those provided by any other packet scheduler [11, 18]. Each packet arriving to an EDF scheduler is assigned a deadline equal to the sum of its arrival time and associated delay bound, and queued packets are transmitted in increasing order of deadline. Since an EDF scheduler selects packets for transmission according to their deadlines, its implementation requires sorting mechanisms. However, the high overhead costs of sorting prohibit the use of EDF in high-speed networks. For this reason, approximate scheduling disciplines with simpler implementations that achieve an efficiency similar to EDF are needed.

In this paper we design, analyze, and evaluate a novel packet scheduling method that approximates EDF, the *Rotating-Priority-Queues*$^+$ (RPQ$^+$) scheduler. We will demonstrate that RPQ$^+$ is a near-optimal packet scheduler in the sense that it can approximate the efficiency of the optimal EDF scheduler with arbitrary precision. The RPQ$^+$ scheduler does not require sorting but rather inserts packets into prioritized FIFO queues; the priorities of these FIFO queues are changed (*rotated*) periodically to increase the priority of waiting packets over time. In switches with shared-memory output buffers, the RPQ$^+$ queue rotation can be implemented efficiently through pointer manipulation. The RPQ$^+$ scheduler has the following three key characteristics: (1) The operations of RPQ$^+$ are independent of the number of queued packets. (2) The RPQ$^+$ scheduler can provide worst-case delay guarantees. (3) RPQ$^+$ always yields a higher network utilization than the *Static Priority* (SP) scheduler which does not change the priorities of queues. So far no existing packet scheduler that tries to approximate EDF can satisfy all of the above characteristics (See Section 3). We derive necessary and sufficient conditions for schedulability in RPQ$^+$, that is, conditions for which all packets are guaranteed to be transmitted at or before their delay bounds.

Using these conditions, we demonstrate that when the rotation period is infinite, i.e., the FIFO queues are never rotated, the efficiency of RPQ$^+$ is identical to SP. We then show that increasing the frequency of queue rotations always yields a higher efficiency, converging to the efficiency of EDF in the limit. We note, however, that greater efficiency requires additional computational overhead in terms of added FIFO queues and more frequent rotations. We compare the efficiency of RPQ$^+$ against other packet schedulers using empirical examples, including an example based on MPEG-compressed video traces [9].

The remainder of this paper is structured as follows. After discussing the framework of a bounded-delay service and related work in Sections 2 and 3, respectively, we describe the RPQ$^+$ scheduler in Section 4. In Section 5 we discuss a shared-memory implementation of RPQ$^+$, and we compare the operational overhead of RPQ$^+$ with other packet schedulers that approximate the efficiency of EDF. In Section 6 we derive necessary and sufficient conditions for schedulability in RPQ$^+$ and show that RPQ$^+$ is a hybrid between SP and EDF. We finally evaluate RPQ$^+$ in Section 7 using numerical examples as well as MPEG-compressed video traces.

2 Framework for Bounded-Delay Services

We consider connection-oriented packet-switched networks where all packets on a single connection traverse the network on a fixed path of switches and links. A client desiring a new connection submits to the network (1) a *traffic characterization* that specifies the maximum traffic on the connection and (2) the desired delay bound [7]. The network uses these specifications in *admission control tests* to ensure the availability of network resources along the path of the connection. In this section we review traffic characterizations as well as packet schedulers and their admission control tests.

2.1 Traffic Characterization

Since a bounded-delay service provides worst-case guarantees, traffic characterizations that we consider must provide a worst-case bound for the maximum traffic arrivals on a connection. Let $\mathcal{C}$ denote the set of connections at a packet scheduler. For a given connection $i \in \mathcal{C}$, let $A_i[\tau, \tau+t]$ denote the total traffic on the connection in time interval $[\tau, \tau+t]$ measured in terms of transmission time of the link. A *traffic constraint function* A_i^* provides a time-invariant bound on A_i as follows [5]:

$$A_i^*(t) \geq A_i[\tau, \tau+t] \qquad \forall t \geq 0, \forall \tau \geq 0 \tag{1}$$

In practice, traffic constraint functions for a connection are expressed with a small number of parameters using a *traffic model* [7, 13, 16, 18, 20, 25, 27, 29]. As an example, we consider the traffic constraint function A^* for the (σ, ρ) traffic model [5] that corresponds with the traffic admitted by a leaky bucket mechanism [24]. The (σ, ρ) model uses a burstiness parameter σ and a rate

Packet Scheduler	Schedulability Conditions
EDF	For all $t \geq d_1$: $$t \geq \sum_{i \in \mathcal{C}} A_i^*(t - d_i) + \max_{d_j > t} s_j^{max}$$
SP	For all p and all $t \geq 0$, there exists a $0 \leq \tau \leq d_p - s^{min}$ such that: $$t + \tau \geq \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(t+\tau) + \sum_{i \in \mathcal{C}_p} A_i^*(t) - s^{min} + \max_{q > p} s_q^{max}$$

Table 1: Necessary and sufficient schedulability conditions.

parameter ρ, and its worst-case traffic is given as follows [5]:

$$A_i^*(t) = \sigma + \rho t \qquad \forall t \geq 0 \tag{2}$$

2.2 Packet Schedulers and Admission Control

Packets from different connections routed on a single outgoing link of a packet switch are stored in a transmission queue, and a packet scheduler determines the transmission order of these packets. Each connection i has a local delay bound d_i at a switch, and a packet on connection i arriving to the switch at time t is assigned a *deadline* of $t + d_i$. A *deadline violation* occurs if any packet is not fully transmitted before its deadline. Since propagation and processing delays are largely fixed due to physical constraints, we assume for clarity of presentation that these delays are zero, and so d_i is a bound on the sum of the queueing delay and transmission time. In the remainder of the paper we restrict our attention to the delay at a single network switch. The case of multi-hop routes can be addressed by either quantifying the distortion of the worst-case traffic arrivals A_i^* at different switches [6] or controlling the distortion of the arrivals by reshaping the traffic to conform to A_i^* at each switch with so-called traffic shaping mechanisms [28].

For a given scheduler, we say that a set $\mathcal{C}$ of connections with traffic constraint functions and delay bounds $\{A_i^*, d_i\}_{i \in \mathcal{C}}$ is *schedulable* if a deadline violation cannot occur for any connection i that conforms its traffic to A_i^* as shown in equation (1). The conditions which determine if a set of connections is schedulable, called *schedulability conditions*, constitute the admission control test in bounded-delay services.

Table 1 shows necessary and sufficient schedulability conditions for the EDF and SP packet schedulers that are derived in [18]. In the EDF condition we use s_j^{max} to denote the maximum transmission time of a packet from connection j. In the SP condition the set of connections $\mathcal{C}$ is partitioned into P connection sets $\{\mathcal{C}_p\}_{1 \leq p \leq P}$, and all connections in $\mathcal{C}_p$ have the same delay bound d_p. We use s_p^{max} to denote the maximum transmission time of any packet from a connection in set $\mathcal{C}_p$, while we use s^{min} to denote the minimum packet transmission time of any packet.

3 Related Work

Recently, several packet schedulers have been considered that approximate EDF with simple implementations [17, 19, 21, 22]. Recall that the main drawback of implementing an EDF scheduler is the sorting operation needed to order packets according to their deadlines. For implementations that use a sorted transmission queue, the complexity of inserting a new packet into the queue is $O(\log N)$, where N is the number of queued packets. At high transmission rates the number of queued packets can be large and the overhead of EDF scheduling can be prohibitive. The approaches in [17, 19, 21, 22] avoid the sorting operation using a similar set of mechanisms. First, all schedulers employ a set of prioritized FIFO queues. Second, each FIFO contains only packets with *laxities* in a certain range, where the laxity of a packet is the time remaining before its deadline. Finally, all schedulers partition the set of connections $\mathcal{C}$ into P connection sets $\{\mathcal{C}_p\}_{1 \leq p \leq P}$ where all connections in $\mathcal{C}_p$ have the identical delay bound d_p.

We first review in Section 3.1 the HOL-PJ scheduler presented in [19, 22] that inserts a packet into a FIFO based on its deadline and subsequently moves individual packets to higher-priority FIFOs as dictated by their laxities. In Sections 3.2 and 3.3 we discuss the priority relabeling architecture [21, 22] and the RPQ scheduler [17], respectively. These packet schedulers are distinct from the first approach in that they do not move individual packets between queues. Instead, they use so-called *calendar queues* [3, 15] that relabel FIFOs periodically to increase the priorities of queued packets.

3.1 Head-of-Line with Priority Jumps (HOL-PJ)

Lim and Kobza present in [19] the Head-of-Line with Priority Jumps (HOL-PJ) scheduler. HOL-PJ maintains P FIFO queues labeled FIFO 1, FIFO 2, ..., FIFO P, and FIFO q has associated laxity range $[d_{q-1}, d_q]$ with $d_0 = 0$. An arriving packet with delay bound d_q is inserted into FIFO q. To keep packets in the appropriate queues, HOL-PJ maintains a timer for each FIFO queue. The timer for FIFO q expires when the first packet in FIFO q violates the laxity of this queue. Then the packet is dequeued and inserted into FIFO $(q-1)$. A generalization of HOL-PJ called the *recirculation architecture* is presented in [22].

Note that HOL-PJ is an exact implementation of EDF. HOL-PJ has advantages over straightforward implementations of EDF with a single transmission queue in that inserting and removing packets can be performed independent of the number of queued packets. However, HOL-PJ has drawbacks in that it requires a large number of timers and necessitates copying packets between different FIFO queues. Note that the copying of packets can be avoided in shared-memory switches in which FIFO queues are implemented as linked lists.

3.2 Priority Relabeling Architecture

In the *priority relabeling architecture* presented by Peha and Tobagi [21, 22], supported delay bounds are of the form $d_p = p\Delta$ for $1 \leq p \leq P$, where Δ is a parameter of the scheduler. The maximum delay bound supported by the priority relabeling architecture is $P\Delta$. As in HOL-PJ, packets arriving with delay bound d_p are placed into FIFO p. Every Δ time units, the priority relabeling architecture modifies the priorities of the FIFOs by relabeling FIFO p as FIFO $(p-1)$ for all $1 < p \leq P$. The laxity range of FIFO p is $[d_{p-1}, d_{p+1}]$ for $1 \leq p < P$ and $[d_{P-1}, d_P]$ for FIFO P. Packets that reside in FIFO 1 during such a relabeling are considered as a special case; either (1) all packets in FIFO 1 are dropped, or (2) FIFO 1 and FIFO 2 are concatenated to form the new FIFO 1. Although [22] recommends the former choice, i.e., dropping packets in FIFO 1, for services in which late packets are to be dropped, observe that the scheduler may drop packets that have not violated their deadlines.

As compared to HOL-PJ, the priority relabeling architecture has a much simpler implementation since it requires only a single timer and does not require the movement of queued packets. The relabeling of FIFOs can be accomplished by simply altering an offset in the priority selector [22], and the additional implementation overhead as compared to an SP scheduler is in the relabeling of priorities. To avoid copying packets during the concatenation of FIFO 1 and FIFO 2, the FIFOs must implemented as linked lists in shared memory. Note also that the priority relabeling architecture is not appropriate for bounded-delay services since schedulability conditions are not available and the scheduler may prematurely drop packets in FIFO 1 that have not violated their deadlines.

3.3 Rotating-Priority-Queues

The *Rotating-Priority-Queues* (RPQ) scheduler presented in [17] is an approximation of EDF designed to be used with physically separated FIFO buffers that does not require a shared memory. RPQ is similar to the priority relabeling architecture described above in that it supports P delay bounds of the form $d_p = p\Delta$. RPQ maintains $P+1$ FIFO queues with indices $0, 1, \ldots, P$, and every Δ time units the FIFOs are relabeled during a so-called *queue rotation*: FIFO p is relabeled as FIFO $(p-1)$ for $p \geq 1$ and FIFO 0 is relabeled as FIFO P. FIFO 0 is included to hold packets from FIFO 1 that have not violated their deadlines at the time of a queue rotation. Arriving packets are never inserted directly into FIFO 0.

In [18] necessary and sufficient schedulability conditions are derived for RPQ that guarantee the transmission of all packets before their deadlines, and we next state these conditions. Let s_p^{max} denote the maximum transmission time of a packet from $\mathcal{C}_p$. All packets from a set of connections $\mathcal{C}$ will be transmitted prior to their deadlines if and only if the following condition holds for all $t \geq$

d_1 [18]:

$$t \geq \sum_{i \in \mathcal{C}_1} A_i^*(t - d_1) + \sum_{p=2}^{P} \sum_{i \in \mathcal{C}_p} A_i^*(t - d_p + \Delta) + \max_{d_q > t - \Delta} s_q^{max} \tag{3}$$

A comparison of the condition for RPQ in equation (3) with the EDF condition in Table 1 shows that RPQ can approximate EDF with arbitrary precision if Δ is selected sufficiently small. Note that the schedulability conditions guarantee that FIFO 0 is empty at queue rotations.

Reducing Δ should result in higher efficiency at the expense of higher overhead costs due to more frequent priority relabeling. However, the efficiency of RPQ may be lower than SP for some choices of Δ. In [18] we presented the following pathological example where RPQ cannot admit connections that are admissible by both EDF and SP. Consider two connection sets 1 and 2 with delay bounds $d_1 = 10$ms and $d_2 = 20$ms that have identical traffic constraint functions A_i^* $(i = 1, 2)$:

$$A_i^*(t) = \left\lfloor \frac{t}{20} \right\rfloor + 1 \tag{4}$$

For each packet scheduler, we compute restrictions on N_1 and N_2, the number of admissible connections of type 1 and 2, respectively. Using the schedulability conditions for EDF and SP in Table 1, one can obtain the following identical restrictions for both EDF and SP: $N_1 \leq 9$ and $N_1 + N_2 \leq 20$. Similarly, one obtains restrictions for RPQ scheduling using the condition in equation (3): $N_1 \leq 9$ and $N_1 + N_2 \leq 20 - \Delta$. Note that the only finite choice of Δ for which an RPQ scheduler supports the same number of connections as EDF and SP is $\Delta = 0$. In the next section, we present a novel scheduling discipline that addresses the above problem while also retaining many of the desirable properties of the RPQ.

4 The Rotating-Priority-Queues$^+$ (RPQ$^+$) Scheduler

In this section we introduce the *Rotating-Priority-Queues*$^+$ (RPQ$^+$) scheduler that approximates the optimal EDF scheduler. Similar to the scheduling disciplines described in the previous section, RPQ$^+$ can be implemented with a set of prioritized FIFO queues that are relabeled periodically. The efficiency of SP provides a lower bound on that of RPQ$^+$, and the efficiency of RPQ$^+$ increases with the frequency of relabeling, approaching that of EDF in the limit. In a shared-memory architecture where FIFO queues are implemented as linked lists, we demonstrate that RPQ$^+$ has low overhead costs that are appropriate for use in high-speed networks. Here we describe the operations of an RPQ$^+$ scheduler and illustrate its operations using a simple example.

4.1 RPQ$^+$ Scheduling

Connections submitting traffic to an RPQ$^+$ scheduler are partitioned into P disjoint connection sets $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_P$, and all connections in $\mathcal{C}_p$ have identical delay bounds $d_p = p\Delta$, where Δ is the rotation interval.

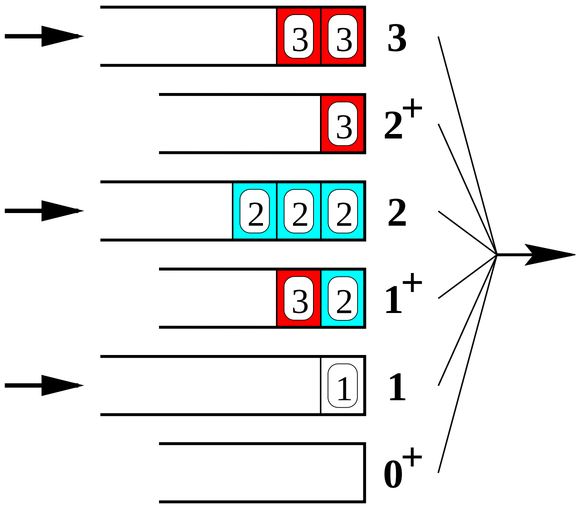


Figure 1: RPQ$^+$ scheduler.

The RPQ$^+$ scheduler employs $2P$ ordered FIFO queues, and these FIFOs are indexed as follows, from highest to lowest priority: $0^+, 1, 1^+, 2, 2^+, \ldots, (P-1), (P-1)^+, P$. We refer to the FIFO with index p (p^+) as FIFO p (FIFO p^+). The RPQ$^+$ scheduler always selects a packet from the highest-priority nonempty FIFO for transmission. All packets arriving on a connection in set $\mathcal{C}_p$ are placed in FIFO p. Arriving packets are never placed directly into FIFO p^+ for any p.

Similar to RPQ, the FIFO queues for an RPQ$^+$ scheduler are relabeled every Δ time units. A RPQ$^+$ queue rotation can be viewed as a two-step process: a so-called "concatenation step" and a so-called "promotion step." In the concatenation step, the current FIFO p and FIFO p^+ are merged to form FIFO p for all $1 \leq p < P$. Packets from FIFO p^+ are concatenated to the end of those from FIFO p. In the promotion step, FIFO p is relabeled as the FIFO $(p-1)^+$ for all $1 \leq p \leq P$. Also, a new empty FIFO p is created for all p to hold packet arrivals during the next rotation interval. After the promotion step, all packets reside in some FIFO p^+.

4.2 Illustration of RPQ$^+$ Scheduling

The operations of an RPQ$^+$ scheduler are best illustrated by means of a simple example. Figure 1 shows an RPQ$^+$ scheduler that supports three connection sets $\mathcal{C}_1, \mathcal{C}_2$, and $\mathcal{C}_3$ with delay bounds $d_p = p\Delta$ for $p = 1, 2, 3$. Packets from connection set $\mathcal{C}_p$ are labeled p. An RPQ$^+$ scheduler that supports these connection sets requires 6 FIFO queues with indices $\{0^+, 1, 1^+, 2, 2^+, 3\}$. FIFO p^+ is indented for all p to distinguish these queues from FIFO p. In Figure 1 packets are assumed to arrive from the left, and packets from connection set $\mathcal{C}_p$ are placed into FIFO p. When a packet is selected for transmission, it is assumed to leave the scheduler at right. If packets are in the scheduler as shown in Figure 1, the next packet selected will be the packet from connection set $\mathcal{C}_1$ since FIFO 1 is the highest-priority nonempty queue.

Figure 2 illustrates queue rotations and scheduling operations for the RPQ$^+$ scheduler over the course of three rotation intervals. Assuming that the scheduler begins operation at time 0, Figure 2(a) shows, from left to right, (i) the state of the queues before the first queue rotation at time Δ, (ii) the concatenation step of the queue rotation, and (iii) the promotion step of the queue rotation. The concatenation step shown in Figure 2(a)(ii) involves merging FIFO p and FIFO p^+ into a single FIFO p for all $1 \leq p < P$. We indicate the queues to be merged with dashed lines at the left of the figure. Figure 2(a)(iii) shows the promotion of packets from FIFO p to FIFO $(p-1)^+$ for $p = 1, 2, 3$. Note that three new queues FIFO p are included in Figure 2(a)(iii) for new arrivals during the next rotation interval.

Figure 2(b)(i) depicts the state of the queues at time 2Δ. In $[\Delta, 2\Delta)$, packet arrivals from connection set $\mathcal{C}_p$ are placed into FIFO p, but packets from the same connection set that arrived during the previous rotation interval reside in FIFO $(p-1)^+$. The second queue rotation at time 2Δ is illustrated in Figures 2(b)(ii) and 2(b)(iii). In the concatenation of FIFO p and FIFO p^+ for $1 \leq p < P$, shown in Figure 2(b)(ii), all packets from FIFO p^+ are inserted at the tail of FIFO p. Figure 2(b)(iii) shows the promotion step of the queue rotation.

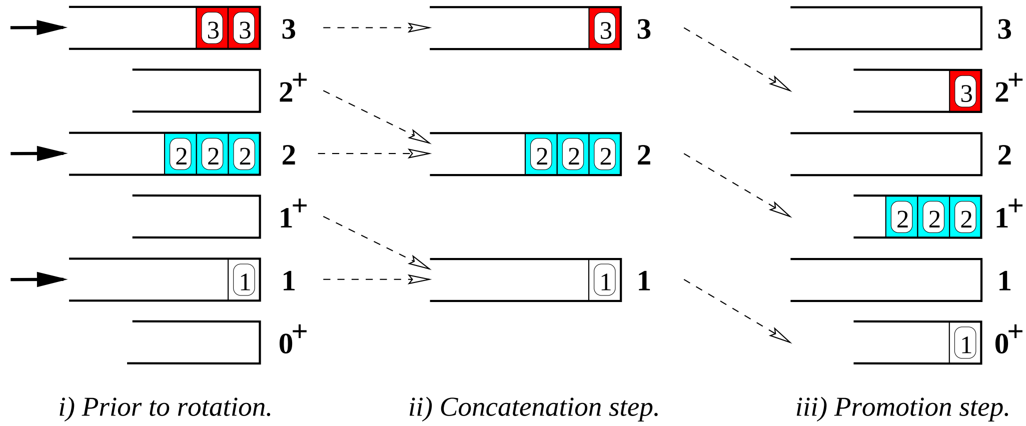
Figures 2(c)(i) depicts the RPQ$^+$ scheduler at time $3\Delta^-$, and Figures 2(c)(ii) and 2(c)(iii) illustrate the two phases of the queue rotation at time 3Δ. Note in Figure 2(c)(iii) that packets from all 3 connection sets are moved to the highest-priority FIFO 0^+ at time 3Δ.

Observe that we do not specify a location to which packets in FIFO 0^+ are moved during a queue rotation. This problem is not of concern if RPQ$^+$ is used in a bounded delay service where all packets are guaranteed to be transmitted before their deadlines. In this case, the delay bounds for each connection set are selected such that FIFO 0^+ will necessarily be empty at the end of each rotation interval. However, for services other than a bounded-delay service, RPQ$^+$ can be designed to either discard all packets in FIFO 0^+ since they have necessarily violated their deadlines or leave these packets in FIFO 0^+ and concatenate new packets to the end of the FIFO.

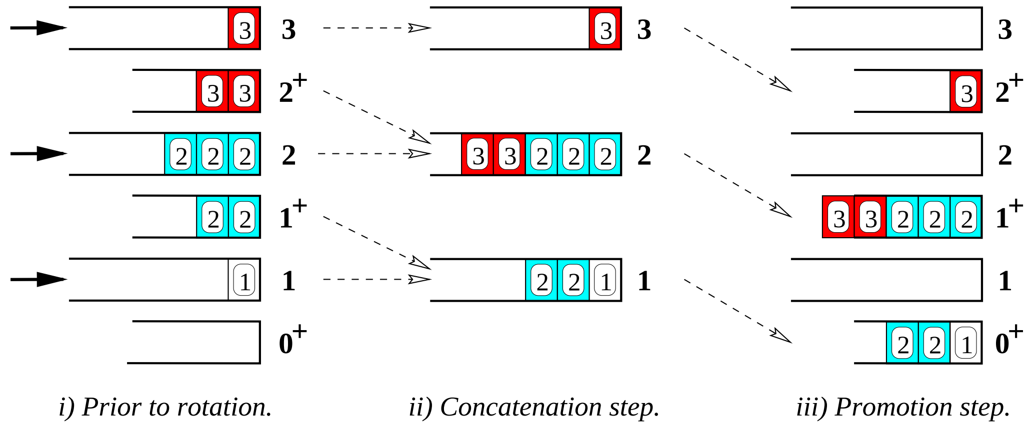
5 Implementation Issues

Here we investigate the operations required for implementing the RPQ$^+$ queue rotation and demonstrate that RPQ$^+$ is feasible for use in high-speed networks. The overhead for implementing RPQ$^+$ is identical to that of an SP scheduler except for the queue rotations. In switches that use shared-memory output buffering, queue rotations can be implemented with a small number of operations using simple pointer manipulations, meaning that RPQ$^+$ requires little overhead when compared with SP.

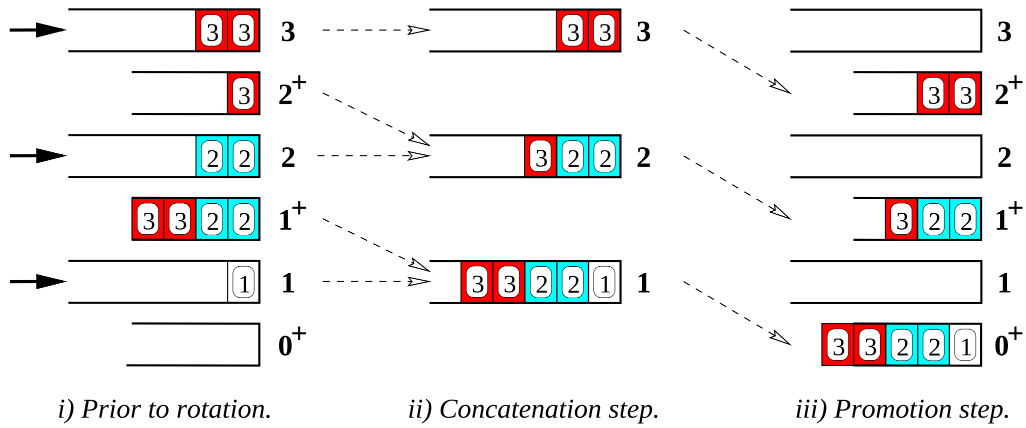
In a switch with output buffering, arriving packets are passed through the switching fabric and then buffered until they are selected for transmission on an outgoing link. On a per-port or per-connection basis, the output buffer memory consists of either a single shared memory pool or physically separate memory. Most ATM switches use output buffers, and the majority of these are



(a) Rotation at time Δ.



(b) Rotation at time 2Δ.



(c) Rotation at time 3Δ.

Figure 2: Example of RPQ$^+$ scheduling operations and queue rotations.

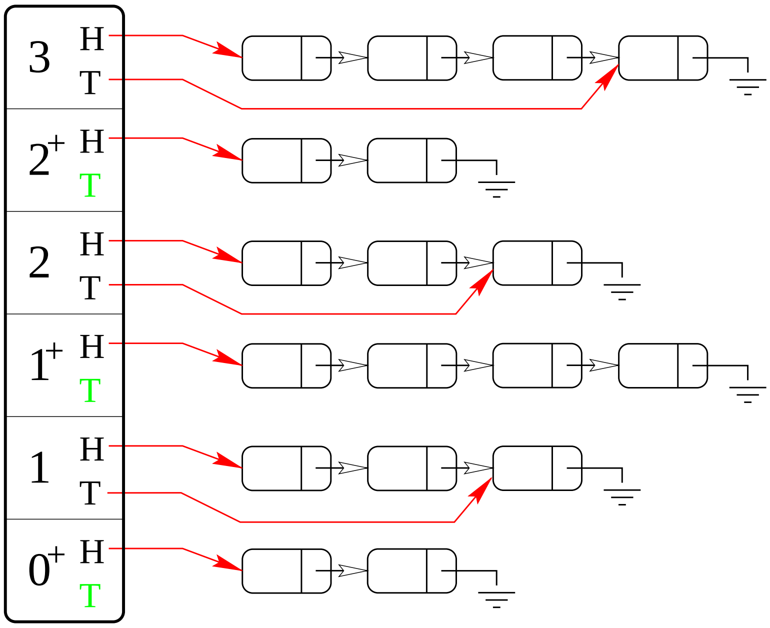


Figure 3: Shared-memory output buffer management in RPQ$^+$.

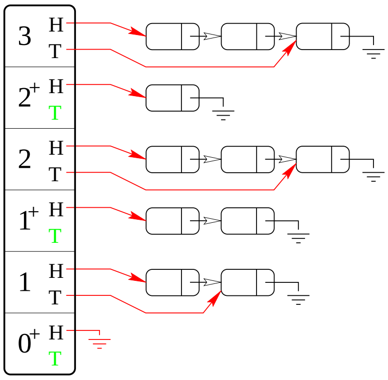
shared-memory buffers [1, 2, 10]. We assume that RPQ$^+$ operates in switches that have shared-memory output buffers. In such a switch, the FIFO queues of RPQ$^+$ can be implemented using linked lists.

Figure 3 shows buffer management for an outgoing link that employs the RPQ$^+$ scheduler. The figure illustrates FIFO queues that are each implemented as a linked list. For each FIFO queue, a "H" and a "T" pointer refer to the head and the tail of the queue, respectively. We use $H(.)$ to denote the head of a queue and $T(.)$ for the tail of the queue; for example, $H(1^+)$ refers to the head of FIFO 1^+. Packets arriving to FIFO p are inserted at $T(p)$, and the scheduler selects packets for transmission from FIFO p at its head $H(p)$.[1]

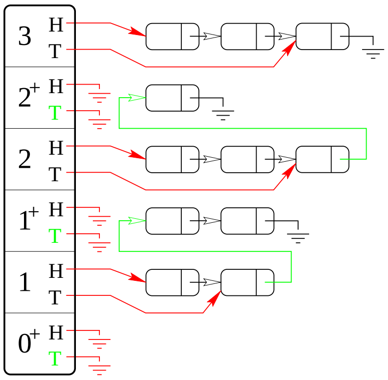
We now consider the complexity of implementing an RPQ$^+$ queue rotation which we illustrate in Figure 4. Figure 4(a) shows the state of an RPQ$^+$ scheduler immediately before a queue rotation, while Figures 4(b) and 4(c) illustrate the concatenation and promotion phases, respectively, of the RPQ$^+$ queue rotation. The operations for the queue rotation involve simple pointer manipulations, and the number of memory accesses in a queue rotation largely determines the total time required. Notice in Figure 4(a) that we assume FIFO 0^+ to be empty immediately prior to a queue rotation, an assumption that holds in a bounded-delay service where all packets are transmitted before their deadlines.

Figure 4(b) depicts all pointer manipulations required for the concatenation phase of queue rotation. FIFO p and FIFO p^+ are concatenated by linking the tail of FIFO p to the head of

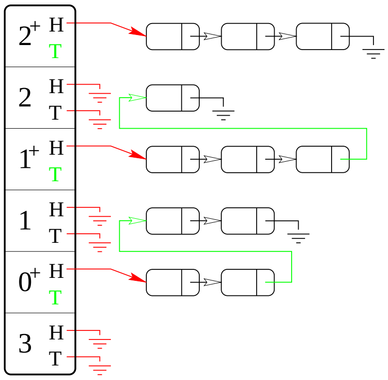
[1]Note that $T(p^+)$ is not used since arriving packets are never inserted into FIFO p^+ for any p.



(a) Immediately before rotation.



(b) After concatenation step.



(c) After promotion step.

Figure 4: Implementation of RPQ$^+$ queue rotation.

the former FIFO p^+. For each concatenation four memory accesses are required: one to obtain the $T(p)$, one to obtain the $H(p^+)$, and two to modify and store the new pointer value. Also illustrated in the figure, the pointers $H(p^+)$ and $T(p^+)$ must be cleared, requiring two memory accesses for each of these P FIFOs.

Figure 4(c) shows that the promotion phase of RPQ$^+$ can be implemented by simply offsetting priorities. Here FIFO p is relabeled as FIFO $(p-1)^+$, FIFO p^+ is relabeled as FIFO p, and FIFO 0^+ is relabeled as FIFO P. This phase of the queue rotation is implemented exactly the same as the rotation in RPQ and the priority relabeling architecture. For this reason, the pointer manipulations illustrated in Figure 4(b) comprise the additional overhead of RPQ$^+$ as compared to RPQ.

6 RPQ$^+$ Schedulability Conditions

In this section we present schedulability conditions for the RPQ$^+$ scheduler and show that its efficiency is always superior to that of SP. We first derive an expression for the workload transmitted before an arbitrary packet in RPQ$^+$ for the general class of traffic constraint functions described in Section 2.1. This expression is central to proving the schedulability conditions and also enables an intuitive understanding of the schedulability conditions. We then present in Theorem 1 the exact, that is, necessary and sufficient, schedulability conditions for a general class of traffic constraint functions. We finally show explicitly that RPQ$^+$ is superior to SP and that its efficiency increases monotonically for a certain class of rotation intervals.

6.1 Workload Transmitted before an Arbitrary Packet

Assume without loss of generality that a packet from connection j in connection set $\mathcal{C}_p$ arrives to an RPQ$^+$ scheduler at time t. We further assume that the packet is fully transmitted by the scheduler at time $t+\delta$. Here we derive an expression $W^{p,t}(t+\tau)$ that represents the total transmission time of all traffic in the scheduler at time $t + \tau$ to be transmitted before the tagged packet.

The tagged packet arriving at time t arrives after a queue rotation that occurred at time $t - \tau_\Delta$, where $0 \leq \tau_\Delta < \Delta$. Queue rotations occur every Δ time units, and so we can express queue rotation times in terms of τ_Δ as follows:

$$\{(t - \tau_\Delta) + i\,\Delta \mid i \text{ an integer}\} \tag{5}$$

Let us consider an arbitrary connection set $\mathcal{C}_q$ and determine the times for which packet arrivals from connections $j \in \mathcal{C}_q$ have higher priority than the tagged packet. We consider three cases: connections from the same connection set as the tagged packet ($q = p$), connections of higher priority than the tagged packet ($q < p$), and connections of lower priority ($q > p$).

(a) $\mathbf{q} = \mathbf{p}$: Since all packets from connection set $\mathcal{C}_p$ are transmitted in FIFO order, all packets that arrive before time t are transmitted before the tagged packet. The interval corresponding with $\mathcal{C}_p$ is $[0, t]$.

(b) $\mathbf{q} < \mathbf{p}$: For a higher-priority connection set $\mathcal{C}_q$ with $q < p$, the packets transmitted before the tagged packet are those that arrive before the instant the tagged packet is rotated into FIFO $(q-1)^+$. The tagged packet will be moved to FIFO $(p-1)^+$ during the first queue rotation after t, i.e., time $(t - \tau_\Delta) + \Delta$, and it will be moved to FIFO $(p-2)^+$ at time $(t - \tau_\Delta) + 2\Delta$. Thus, at time $(t - \tau_\Delta) + 2\Delta$ the tagged packet will be moved into a FIFO with higher-priority than FIFO $(p-1)$, the FIFO into which from connection set $\mathcal{C}_{p-1}$ are placed. More generally, at time $(t - \tau_\Delta) + (n+1)\Delta$ the tagged packet will have a higher priority than new arrivals from connection set $\mathcal{C}_{p-n}$ for $n \geq 1$. Taking into account that the packet departs the scheduler at time $t + \delta$, the time interval corresponding with connection set $\mathcal{C}_q$ for $q < p$ is given by $[0, \min\{t + \delta, (t - \tau_\Delta) + (p - q + 1)\Delta\}]$.

(c) $\mathbf{q} > \mathbf{p}$: For lower-priority connection sets $\mathcal{C}_q$ with $q > p$, only packets that have been rotated into some FIFO r^+ with $r < p$ will be transmitted before the tagged packet. Consider for example packets from connection set $\mathcal{C}_{p+1}$ that arrive up until time $(t - \tau_\Delta) - \Delta$. At time $(t - \tau_\Delta) - \Delta$, these packets will be moved from FIFO $(p+1)$ to FIFO p^+, and they will subsequently be moved to FIFO $(p-1)^+$ at time $t - \tau_\Delta$. Consequently, packets arriving in the time interval $[0, (t - \tau_\Delta) - \Delta]$ will be transmitted before the tagged packet. Note that packets from connection set $\mathcal{C}_{p+1}$ arriving after this time interval will reside in a FIFO with lower priority than FIFO p at time t and hence will not be transmitted before the tagged packet. More generally, for connection set $\mathcal{C}_q$ with $q > p$, packets arriving up until time $(t - \tau_\Delta) + (p - q)\Delta$ will be transmitted before the tagged packet, resulting in the interval $[0, (t - \tau_\Delta) + (p - q)\Delta]$.

The intervals shown above describe the traffic transmitted before the tagged packet, but these intervals do not account for the effects of nonpreemption of packets. In particular, consider a scenario where, at some time prior to the arrival of the tagged packet at time t, there are no packets in the scheduler with arrival times included in the intervals described above. Since the RPQ$^+$ scheduler is work-conserving, some packet not included in the intervals may be transmitted before the tagged packet. We next account for such a nonpreemption in order to accurately quantify the traffic to be transmitted before the tagged packet.

We define $t - \hat{\tau}$ to be the last time before t that the RPQ$^+$ scheduler does not contain packets that are to be transmitted before the tagged packet. Note that such a time is guaranteed to exist since the scheduler is assumed to be empty at time 0. If we use $W_i(\tau)$ to denote the workload in the RPQ$^+$ scheduler from connection $i \in \mathcal{C}$ at time τ, then we can write $\hat{\tau}$ directly from the intervals above as follows:

$$\hat{\tau} = \min\{z \,|\, \sum_{q=1}^{p} \sum_{i \in \mathcal{C}_q} W_i(t-z) + \sum_{q=p+1}^{P} \sum_{i \in \mathcal{C}_q} W_i(\min\{t-z, (t-\tau_\Delta) + d_p - d_q\}) = 0, z \geq 0\} \tag{6}$$

By definition of time $t - \hat{\tau}$, the work transmitted by the RPQ$^+$ scheduler during interval $[t-\hat{\tau}, t+\delta]$ is limited to packets with arrival times during the intervals specified above and the remaining transmission time of some other packet in transmission at time $t - \hat{\tau}$, which we denote by $R(t - \hat{\tau})$.

We are now in a position to explicitly write the workload in the scheduler at time $t + \tau$ that will be transmitted before the packet from connection set $\mathcal{C}_p$ with arrival time t is completely transmitted. This workload is denoted by $W^{p,t}(t+\tau)$ and is given as follows for all τ, $0 \leq \tau \leq \delta$:

$$\begin{aligned} W^{p,t}(t+\tau) \;=\; & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i[t-\hat{\tau}, \min\{t+\tau, (t-\tau_\Delta) + (p-q+1)\Delta\}] + \sum_{i \in \mathcal{C}_p} A_i[t-\hat{\tau}, t] + \\ & \sum_{q=p+1}^{P} \sum_{i \in \mathcal{C}_q} A_i[t-\hat{\tau}, (t-\tau_\Delta) + (p-q)\Delta] + R(t-\hat{\tau}) - (\hat{\tau}+\tau) \end{aligned} \tag{7}$$

The first three terms on the right-hand-side of equation (7) account for the arrival intervals derived previously, while the term $R(t - \hat{\tau})$ is the remaining transmission time of the packet transmitted at time $t - \hat{\tau}$. Since by choice of $\hat{\tau}$ the packet scheduler is continuously backlogged for the entire interval $[t - \hat{\tau}, t + \tau]$, the final term accounts for the total workload transmitted during the interval.

Similar to SP, s_p^{max} denotes the maximum transmission time for packets on a connection from set $\mathcal{C}_p$, while s^{min} denotes the minimum packet transmission time for any packet. We assume that the transmission time of the tagged packet is given by s, where $s^{min} \leq s \leq s_p^{max}$. Since the tagged packet completes transmission at time $t + \delta$, the packet must begin transmission at time $t + \delta - s$, a time at which the total workload to be transmitted before or with the tagged packet is s, the transmission time of the packet itself. We can determine the departure time $t + \delta$ of the tagged packet using the following equation:

$$\delta = s + \min\{z \,|\, W^{p,t}(t+z) = s, z \geq 0\} \tag{8}$$

6.2 RPQ$^+$ Schedulability Conditions and Properties of RPQ$^+$

Here we present the necessary and sufficient conditions for schedulability in an RPQ$^+$ scheduler. These conditions assume that all connections have traffic constraint functions as described in Section 2.1.

Theorem 1 *A set $\mathcal{C}$ of connections that is given by $\{A_i^*, d_i\}_{i\in\mathcal{C}}$ is RPQ$^+$-schedulable with rotation interval Δ if and only if for all priorities p and for all $t \geq 0$ there exists a τ with $0 \leq \tau \leq d_p - s^{min}$ such that:*

$$t + \tau \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(\min\{t+\tau, t+d_p-d_q+\Delta\}) + \sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t+d_p-d_q) - s^{min} + \max_{r, d_r > t+d_p} s_r^{max} \quad (9)$$

Due to space limitations, here we include only a sketch of the proof of sufficiency of Theorem 1. A complete proof of the theorem is provided in [26].

Proof Idea for Sufficiency of Theorem 1:
Assuming that the inequality in equation (9) holds for all times $t \geq 0$, we must show that an arbitrary packet from connection $j \in \mathcal{C}_p$ arriving at time t does not violate its deadline.

Starting with equation (7), we provide an upper bound on the workload transmitted before the tagged packet. In the worst case, $\tau_\Delta = 0$, the traffic $A_i[t_1, t_2]$ on connection i is at most $A_i^*(t_2 - t_1)$, and $R(t - \hat{\tau})$ is limited to $\max_{r, d_r > \hat{\tau}+d_p} s_r^{max}$. Combining these observations with equations (7) and (9), we can show that there exists some τ ($0 \leq \tau \leq d_p - s^{min}$) such that $W^{p,t}(t+\tau) \leq s^{min}$. Since the transmission time of the packet must be at least s^{min}, the packet will complete transmission at or before time $t + d_p$ as required. □

In order to compare RPQ$^+$ with EDF, we next present a sufficient (but not necessary) schedulability condition for RPQ$^+$ that has a formulation similar to the exact EDF conditions in Table 1. We obtain these conditions directly from Theorem 1 by substituting $\tau = d_p - s^{min}$ in equation (9).

Corollary 1 *Given a set $\mathcal{C}$ of connections that is given by $\{A_i^*, d_i\}_{i\in\mathcal{C}}$, the connections are RPQ$^+$-schedulable with rotation interval Δ if for all priorities p and for all $t \geq d_p$ the following condition holds:*

$$t \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(t - d_q + \Delta) + \sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t - d_q) + \max_{r, d_r > t} s_r^{max} \quad (10)$$

Comparing the condition in equation (10) with the EDF condition, we see that the only difference in the two conditions is the rotation interval Δ. We see that these two conditions become identical in the limit as $\Delta \to 0$, verifying that an RPQ$^+$ scheduler effectively approximates EDF with arbitrary precision.

The RPQ$^+$ scheduler is designed to be a hybrid between SP and EDF in the sense that (1) RPQ$^+$ always achieves an efficiency at least as good as SP, (2) the efficiency of RPQ$^+$ is nondecreasing as the rotation interval Δ is reduced[2], and (3) the efficiency of RPQ$^+$ approaches that of EDF as $\Delta \to 0$. The latter two claims are easy to show; the second holds since the right-hand-side of equation (9) increases with Δ, while the final claim follows from verifying that the RPQ$^+$ condition in equation (10) and the EDF condition from Table 1 are identical for $\Delta = 0$. We conclude this section by arguing that any set of connections schedulable with SP are also schedulable with RPQ$^+$.

Our argument relies on a necessary condition for SP schedulability.

Lemma 1 *If a set $\mathcal{C}$ of connections given by $\{A_i^*, d_i\}_{i\in\mathcal{C}}$ is SP-schedulable, then all priorities p and for all $t \geq 0$ there exists a τ with $\tau \leq d_p - s^{min}$ such that:*

$$t + \tau \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(t+\tau) + \sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t + d_p - d_q) - s^{min} + \max_{r, d_r > t+d_p} s_r^{max} \tag{11}$$

Due to space limitations, we provide only a proof sketch of Lemma 1. A complete proof is given in [26].

Proof Idea:

We prove Lemma 1 by contradiction. We assume that equation (11) is violated for some priority p and time t for all $0 \leq \tau \leq d_p - s^{min}$. We next construct a scenario in which some packet in connection set $\mathcal{C}_u$ ($u \geq p$) must have a deadline violation at or before time $t + d_p$.

Assume that a packet of maximal size from connection $k \in \mathcal{C}_r$, with $d_r > t + d_p$ arrives to an empty scheduler immediately prior to time 0, and at time 0 all connections $i \in \mathcal{C}_q$ with $d_q \leq t + d_p$ submit traffic according to A_i^*, with the exception that for $\mathcal{C}_q$ ($q \geq p$) a packet with transmission time s^{min} is delayed until time $t + d_p - d_q$. All of these packets, which we refer to collectively as the *delayed packets*, have deadlines at time $t + d_p$.

The last of the delayed packets transmitted is on some connection $j \in \mathcal{C}_u$ ($u \geq p$), and we call this packet the *tagged packet*. Note that the tagged packet will necessarily reside in the scheduler at time t. Assuming that the tagged packet begins transmission at time $t + \delta$, the traffic to be transmitted before this packet at time $t + \tau$, $0 \leq \tau \leq \delta$ includes: (a) All traffic from lower-priority connection sets $\mathcal{C}_q$ ($q \geq p$) that arrives up to time $t + d_p - d_q$, i.e., $\sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t + d_p - d_q)$. Note that all of this traffic has arrived to the scheduler by time t. (b) All traffic from higher-priority connection sets $\mathcal{C}_q$ ($q < p$) that arrives to the scheduler up until time $t + \tau$, i.e., $\sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(t + \tau)$. (c) Due to nonpreemption, the packet arriving before time 0 with transmission time $\max_{d_r > t+d_p} s_r^{max}$. Combining the above observations with our assumption, we

[2] Recall that Δ must evenly divide all desired delay bounds since the supported delay bounds are of the form $d_p = p\Delta$ for all p.

find that the tagged packet from connection set $\mathcal{C}_u$ with transmission time s^{min} will not begin transmission until after time $t + d_p - s^{min}$ and will therefore have a deadline violation. □

With Lemma 1, the necessary condition for SP in equation (11) implies the sufficient condition for RPQ$^+$ in equation (9). Thus RPQ$^+$ always achieves an efficiency at least as high as SP, and so we have shown that RPQ$^+$ is a hybrid of SP and EDF.

7 Evaluation

In this section we compare the efficiency of the RPQ$^+$ scheduler against schedulers EDF, SP, and RPQ using empirical examples. Two sets of experiments are included. The first experiment uses traffic policed by leaky buckets, while the second experiment uses traces of MPEG-compressed video. In both experiments we use the most accurate, i.e., necessary and sufficient, admission control tests for each packet scheduler to obtain a precise comparison of the efficiency of the schedulers. We use the schedulability conditions from Theorem 1 for RPQ$^+$, the conditions from Table 1 for EDF and SP, and the condition in equation (3) for RPQ.

7.1 Numerical Example

In the first experiment we compute the *schedulable region* of the packet schedulers for a set of three connection groups using an approach similar to one used in [14, 18]. We vary the traffic rate of each connection group and use a surface plot to illustrate the rates for which all delay bounds are guaranteed. We can compare the efficiencies of different packet schedulers graphically by comparing the volumes of their schedulable regions.

We consider connections supported at a single packet scheduler that transmits packets at 155 Mbps, a rate that corresponds to OC-3. The three connection groups have traffic that conforms to the (σ, ρ) traffic model, with traffic constraint functions A^* of the form:

$$A^*(t) = \sigma + \rho t \tag{12}$$

Table 2 shows the traffic and QoS parameters for all connection groups. For a connection group with index j, the table shows the delay bound d_j at the scheduler as well as the burst σ_j and the range of rates ρ_j. The bursts b_j in Table 2 are given in 53-byte ATM cells. In the example, we vary the rate parameter ρ_j between 10 and 155 Mbps.

We present our results with three sets of graphs in Figures 5-7. In the first two figures, we illustrate the schedulable region for different schedulers using three-dimensional surface plots. We show the transmission rates for which all connection groups are admissible; the volume beneath a curve includes all operating points for which all packets are guaranteed to be transmitted before their deadlines. Note that the axes in these figures have a logarithmic scale. The last figure, Figure 7, is a two-dimensional plot that summarizes all results in a single graph.

	Index j	Delay Bound d_j	Burst σ_j	Rate ρ_j
Low Delay Group	1	12 ms	4000 cells	10-155 Mbps
Medium Delay Group	2	24 ms	2000 cells	10-155 Mbps
High Delay Group	3	36 ms	4000 cells	10-155 Mbps

Table 2: Parameter Set for Scheduler with 155 Mbps Transmission Rate.

Figure 5 shows the schedulability regions for EDF and SP schedulers as well as a reference graph for evaluating the impact of a bounded-delay service. Figure 5(a) shows the schedulable region without delay constraints, i.e., $d_j = \infty$ for all j. In this case the schedulability condition is that the aggregate traffic rate cannot exceed the rate of the transmission link, that is, $\sum_{j=1}^{3} \rho_j < 155$ Mbps. This schedulable region will contain the schedulable region for all other packet schedulers.

In Figures 5(b) and 5(c), we depict the schedulable regions for EDF and SP packet schedulers, respectively. Since EDF is the optimal packet scheduler with respect to number of admissible connections, the region shown in Figure 5(b) will contain the region corresponding to any other packet scheduler. For the parameter sets considered, observe that the schedulable region for EDF is much larger than that for SP as shown in Figure 5(c).

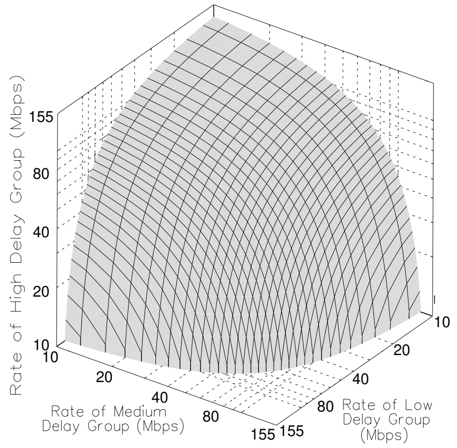
We next show in Figure 6 schedulable regions of the RPQ$^+$ packet scheduler for feasible rotation intervals in the range $\Delta = 1 - 12$ms. For this example, the number of queues that must be maintained for a particular choice of Δ is given by $72/\Delta$. We compare the RPQ$^+$ schedulable regions with the benchmark regions from Figure 5. Note that for all choices of rotation interval Δ, the RPQ$^+$ schedulable region is superior to that of SP in Figure 5(c). Even for $\Delta = 12$ms, the largest possible choice of RPQ$^+$ rotation interval for this example, the RPQ$^+$ schedulable region completely contains the SP region. Notice in Figures 6(a)-(f) that the schedulable region increases as the rotation interval is decreased, closely approximating EDF when $\Delta = 1$ms

Figure 7 summarizes the volumes from the schedulable regions in the previous figures by condensing the information into a single two-dimensional graph. In this graph we show results for the EDF, SP, and RPQ$^+$ schedulers considered in the previous figures, and we also include results for the RPQ scheduler. For a packet scheduler Σ, we let $V^{\Sigma}(\Delta)$ denote the volume of its schedulable region with rotation interval Δ.[3] Letting V^{∞} denote the volume of the schedulable region without deadlines shown in Figure 5(a), we use for evaluation the ratio of $V^{\Sigma}(\Delta)$ and V^{∞} expressed as a percentage, that is:

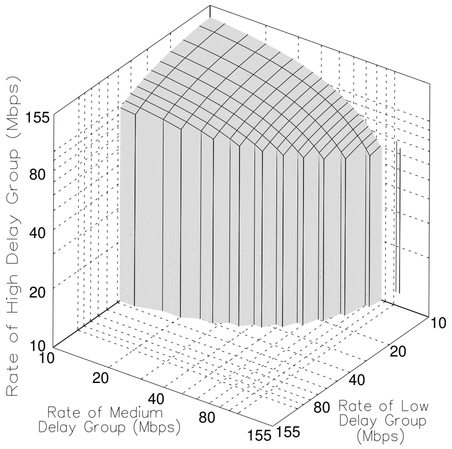
$$\frac{V^{\Sigma}(\Delta)}{V^{\infty}} \cdot 100\%$$

We plot the resulting values for a packet scheduler as a function of Δ. For example, the value 42.1% for EDF in the figure can be interpreted as follows: the volume contained in the EDF schedulable

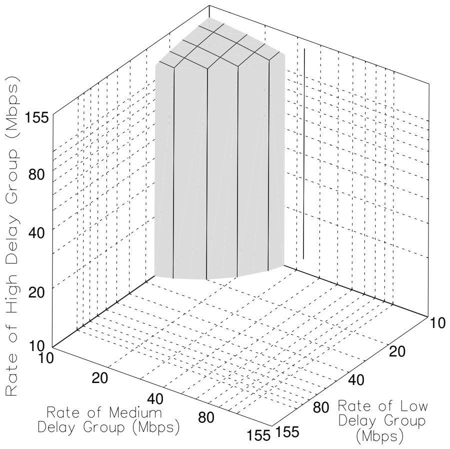
[3] Since the EDF and SP packet schedulers do not employ Δ, both $V^{SP}(\Delta)$ and $V^{EDF}(\Delta)$ are independent of Δ.



(a) Schedulable Region without Delay Constraints.



(b) EDF Scheduler.



(c) SP Scheduler.

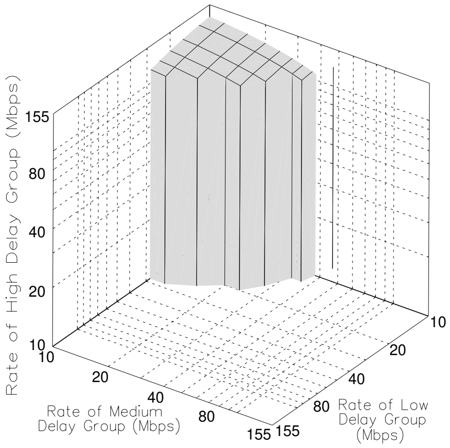
Figure 5: Benchmark Schedulable Regions.

region contains 42.1% of the volume of the region in Figure 5(a).

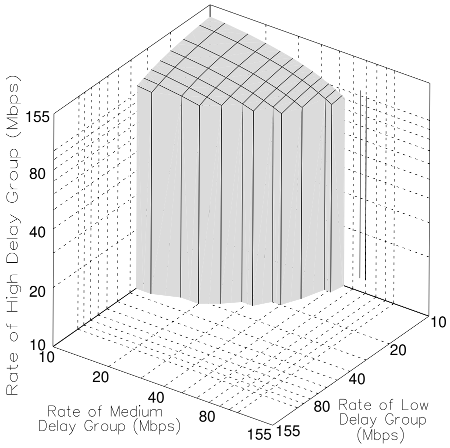
Figure 7 includes all rotation intervals Δ from Figure 6 as well as several additional values. Note in the figure that RPQ$^+$ achieves a schedulability region identical to EDF for $\Delta \leq 0.4$ms, while the RPQ schedulable region is identical to EDF for $\Delta \leq 0.2$ms. Also observe that for this example the RPQ scheduler achieves an efficiency superior to SP only for $\Delta \leq 4$ms.

7.2 MPEG Example

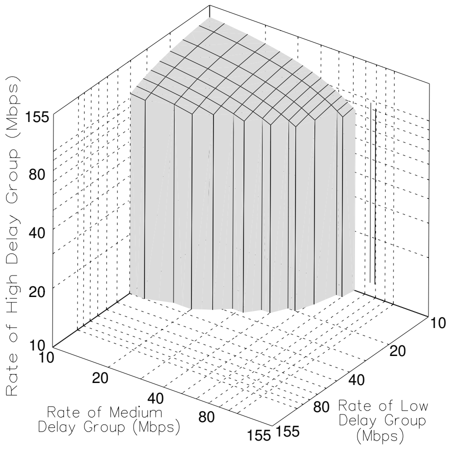
In this experiment all traffic characterizations are obtained from publicly-available traces of MPEG video [23]. We use two MPEG traces for the evaluation: a thirty-minute segment of the James Bond entertainment movie *Goldfinger* (*"Bond"*) and 200 seconds of a video conference recorded



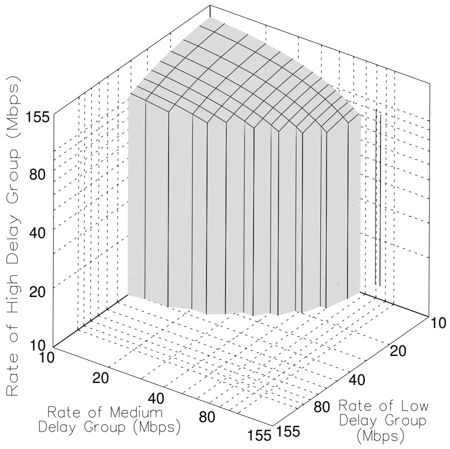
(a) RPQ$^+$ at $\Delta = 12$ms.



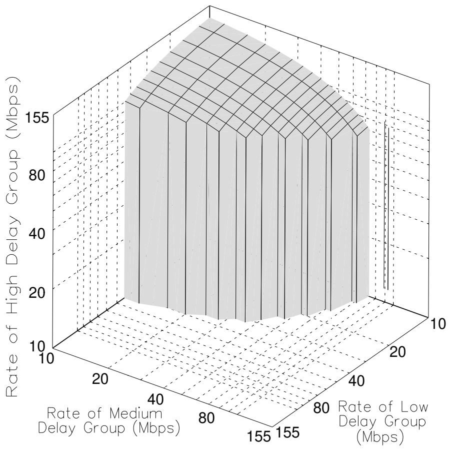
(b) RPQ$^+$ at $\Delta = 6$ms.



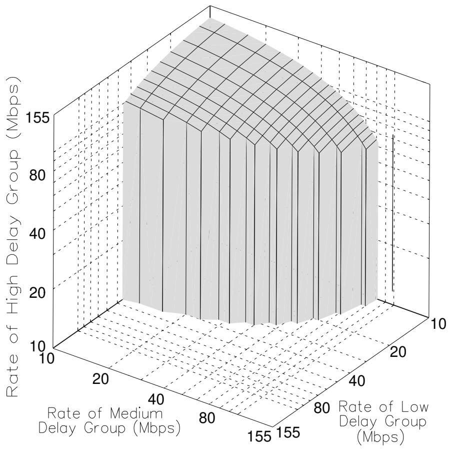
(c) RPQ$^+$ at $\Delta = 4$ms.



(d) RPQ$^+$ at $\Delta = 3$ms.



(e) RPQ$^+$ at $\Delta = 2$ms.



(f) RPQ$^+$ at $\Delta = 1$ms.

Figure 6: Schedulable Regions for RPQ$^+$.

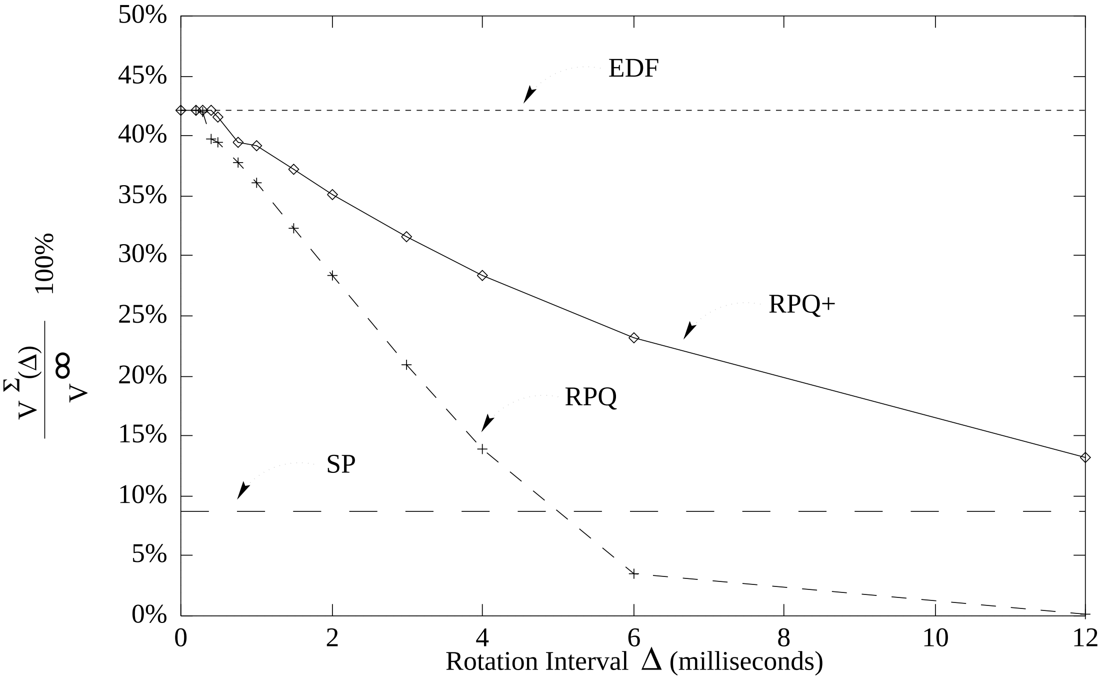


Figure 7: Summary of utilizations for packet schedulers. For each packet scheduler Σ, all values are reported as the ratio of $V^{\Sigma}(\Delta)$, i.e., the volume of the schedulability region of Σ with rotation interval Δ, and V^{∞}, i.e., the volume of the schedulability region as shown in Figure 5(a).

using a set top camera ("*Settop*"). Both traces were encoded in software at 24 frames/second with frame size 384x288 and frame pattern IBBPBBPBBPBB.

We again consider a packet scheduler that operates at 155 Mbps, and we assume that all traffic is packetized in 53-byte ATM cells with a payload of 48 bytes each. We use the so-called *empirical envelope* of a video sequence to characterize its traffic, where the empirical envelope E^* of a sequence with traffic A is given by [4, 25]:

$$E^*(t) = \sup_{\tau \geq 0} A[\tau, \tau + t] \qquad \forall t \geq 0 \tag{13}$$

The empirical envelope is the tightest traffic characterization available for a video sequence and, when used with admission control, will result in the admission of a maximal number of connections. By using empirical envelopes for traffic characterization, we can determine the highest efficiency that can be achieved by a given packet scheduler.

Similar to the previous experiment, we consider two connection groups, one group consisting solely of *Bond* connections, and the second consisting solely of *Settop* connections. All connections in the same group have identical delay bounds: d_{Settop} = 100ms and d_{Bond} = 200ms.

Figure 8 illustrates the number of connections that can be supported at their delay constraints

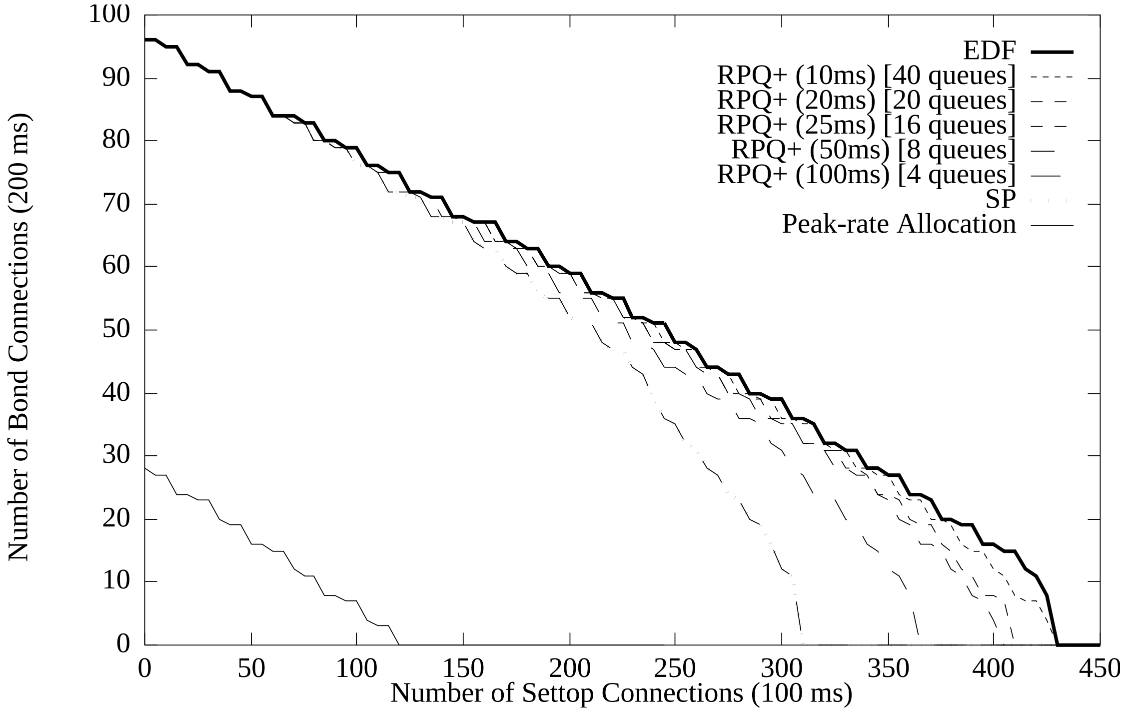


Figure 8: Example of RPQ$^+$ for MPEG video sequences.

for the EDF, SP, and RPQ$^+$ schedulers as well as for a peak-rate allocation scheme.[4] We use a bold solid line for EDF, a bold dotted line for SP, a thin solid line for the peak-rate allocation scheme, and a series of dashed lines for RPQ$^+$ at different values of rotation interval Δ. For the RPQ$^+$ curves we show the value of Δ and the number of required FIFOs. For each packet scheduler, we plot the maximum number of admissible *Bond* connections as a function of the number of *Settop* connections. For example, all packet schedulers (except the peak-rate scheme) can support 96 *Bond* connections if there are no *Settop* connections at the switch, and EDF can simultaneously support 60 *Bond* and 200 *Settop* connections.

We observe in the figure that all of the packet schedulers can admit many more connections than the peak-rate allocation. Additionally, EDF can admit many more *Bond* connections than SP for larger numbers of *Settop* connections. Note that RPQ$^+$ is identical to SP for Δ = 100ms, and smaller values of Δ result in higher efficiency. For Δ = 10ms, a point of operation requiring 40 FIFOs, RPQ$^+$ closely approximates EDF.

[4]The peak rate of a connection is defined as the ratio of the largest-sized frame and the constant interframe time.

8 Conclusions

In this paper we presented the novel Rotating-Priority-Queues$^+$ (RPQ$^+$) packet scheduler. On the one hand, we showed that RPQ$^+$ can achieve a network utilization that approximates that of Earliest-Deadline-First (EDF), the optimal scheduler that can support the tightest delay guarantees in a bounded-delay service. On the other hand, we showed that RPQ$^+$ can be implemented in shared-memory architectures with low overhead costs that make it practical for use in high-speed networks. The RPQ$^+$ scheduler places arriving packets into prioritized FIFO queues based on their delay constraints, and these queues are modified ("rotated") to increase the priority of waiting packets. RPQ$^+$ relies on a so-called rotation interval Δ that determines the frequency of these queue manipulations. When Δ is infinite, i.e., queues are never rotated, RPQ$^+$ behaves exactly the same as a SP scheduler and the two packet schedulers admit the same number of connections; as Δ is reduced, the number of admissible connections increases, approaching that of EDF in the limit. RPQ$^+$ can be implemented similar to SP except for the queue rotations which can be performed by manipulating pointers without moving any queued packets. We presented exact schedulability conditions for RPQ$^+$ that can be used for admission control tests in a bounded delay service. We finally presented experiments that included MPEG traffic sources to illustrate that RPQ$^+$ can closely approximate EDF even for large values of Δ.

References

[1] ForeThough Bandwidth Management Version 1.0, January 1996. Fore Systems White Paper, Available at http://www.fore.com/pdf_files/traffic.pdf.

[2] N. Berry. Asynchronous Transfer Mode (ATM) Switch Technology and Vendor Survey. Technical Report NAS-95-001, National Aeronautics and Space Administration, January 1995.

[3] R. Brown. Calendar Queues: A Fast $O(1)$ Priority Queue Implementation for the Simulation Event Set Problem. *Communications of the ACM*, 21(10):1220–1227, October 1988.

[4] C.-S. Chang. Stability, Queue Length, and Delay of Deterministic and Stochastic Queueing Networks. *IEEE Transactions on Automatic Control*, 39(5):913–931, May 1994.

[5] R. L. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[6] R. L. Cruz. A Calculus for Network Delay, Part II: Network Analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[7] D. Ferrari and D. C. Verma. A Scheme for Real-Time Channel Establishment in Wide-Area Networks. *IEEE Journal on Selected Areas in Communications*, 8(3):368–379, April 1990.

[8] N. R. Figueira and J. Pasquale. An Upper Bound on Delay for the VirtualClock Service Discipline. *IEEE/ACM Transactions on Networking*, 3(4):399–408, August 1995.

[9] D. Le Gall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4):305–313, April 1991.

[10] M. W. Garrett. A Service Architecture for ATM: From Applications to Scheduling. *IEEE Network*, 10(3):6–14, May/June 1996.

[11] L. Georgiadis, R. Guerin, and A. Parekh. Optimal Multiplexing on a Single Link: Delay and Buffer Requirements. In *Proc. IEEE Infocom '94*, pages 524–532, June 1994.

[12] L. Georgiadis, R. Guerin, V. Peris, and K. N. Sivarajan. Efficient Network QoS Provisioning Based on per Node Traffic Shaping. In *Proc. IEEE Infocom '96*, pages 102–110, March 1996.

[13] S. J. Golestani. A Framing Strategy for Congestion Management. *IEEE Journal on Selected Areas In Communications*, 9(7):1064–1077, September 1991.

[14] J. M. Hyman, A. A. Lazar, and G. Pacifici. Real-Time Scheduling with Quality of Service Constraints. *IEEE Journal on Selected Areas in Communications*, 9(7):1052–1063, September 1991.

[15] C. R. Kalmanek, H. Kanakia, and S. Keshav. Rate Controlled Servers for Very High-Speed Networks. In *Proc. Globecom '90*, pages 12–20, December 1990.

[16] E. Knightly and H. Zhang. Traffic Characterization and Switch Utilization Using Deterministic Bounding Interval Dependent Traffic Models. In *Proc. IEEE Infocom '95*, pages 1137–1145, April 1995.

[17] J. Liebeherr and D. E. Wrege. A Versatile Packet Multiplexer for Quality-of-Service Networks. In *Proc. 4th International Symposium on High-Performance Distributed Computing (HPDC-4)*, pages 148–155, August 1995.

[18] J. Liebeherr, D. E. Wrege, and Domenico Ferrari. Exact Admission Control in Networks with Bounded Delay Services. To appear: *IEEE/ACM Transactions on Networking.*

[19] Y. Lim and J. E. Kobza. Analysis of a Delay-Dependent Priority Discipline in an Integrated Multiclass Traffic Fast Packet Switch. *IEEE Transactions on Communications*, 38(5):659–665, May 1990.

[20] A. K. Parekh and R. G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.

[21] J. M. Peha. *Scheduling and Dropping Algorithms to Support Integrated Services in Packet-Switched Networks*. PhD thesis, Stanford University, June 1991.

[22] J. M. Peha and F. A. Tobagi. Implementation Strategies for Scheduling Algorithms in Integrated-Services Packet-Switched Networks. In *Proc. IEEE Globecom '91*, pages 1733–1740, 1991.

[23] O. Rose. Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems. Technical Report 101, Institute of Computer Science, University of Wurzburg, February 1995. The traces used in this paper are available via anonymous ftp from the site ftp-info3.informatik.uni-wuerzburg.de in the directory /pub/MPEG/.

[24] J. S. Turner. New Directions in Communications (or Which Way to the Information Age?). *IEEE Communications Magazine*, 25(8):8–15, October 1986.

[25] D. E. Wrege, E. W. Knightly, H. Zhang, and J. Liebeherr. Deterministic Delay Bounds for VBR Video in Packet-Switching Networks: Fundamental Limits and Practical Tradeoffs. To appear: *IEEE/ACM Transactions on Networking*, June 1996.

[26] D. E. Wrege and J. Liebeherr. RPQ$^+$: a Packet Scheduler for Shared-Memory Architectures. Technical Report CS-96-10, University of Virginia, June 1996.

[27] D. E. Wrege and J. Liebeherr. Video Traffic Characterization for Multimedia Networks with a Deterministic Service. In *Proc. IEEE Infocom '96*, pages 537–544, March 1996.

[28] H. Zhang. Providing End-to-End Performance Guarantees Using Non-Work-Conserving Disciplines. To appear: *Computer Communications*: Special Issue on System Support for Multimedia Computing.

[29] H. Zhang and D. Ferrari. Rate-Controlled Static-Priority Queueing. In *Proc. IEEE Infocom '93*, pages 227–236, April 1993.