A Near-Optimal Packet Scheduler for QoS Networks*

Dallas E. Wrege *Jörg Liebeherr*
Department of Computer Science
University of Virginia
Charlottesville, VA 22903

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 derive admission control tests for RPQ$^+$ and show that it has the following desirable properties: its implementation requires operations independent of the number of queued packets, it can provide worst-case delay guarantees, and its efficiency is "between" that of EDF and Static-Priority (SP) schedulers. 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.

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. Continuous media applications such as voice and video are 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 (See [20] for a recent survey). The well-known *Earliest-Deadline-First* (EDF) scheduler has been studied in [6, 9, 12] 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 [9, 12]. 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 maintaining a sorted queue prohibit the use of EDF in high-speed networks. For this reason, we seek approximate scheduling disciplines with simpler implementations that can achieve an efficiency similar to EDF.

Recently, several packet schedulers have been designed to approximate the efficiency of EDF [11, 13, 15]. Unlike EDF, these approximation schedulers do not require a sorted transmission queue but rather insert packets into a set of prioritized FIFO queues. In each of these schedulers, a given FIFO contains only packets with *laxities* in a certain range, where the laxity of a packet is the time remaining until it violates its deadline. The *Head-of-Line with Priority Jumps* (HOL-PJ) scheduler moves individual packets between FIFOs according to their laxities [13]. HOL-PJ was shown to be an exact implementation of EDF, but its implementation requires a large number of timers and necessitates copying packets between FIFOs. Two schedulers that avoid manipulating individual packets are the priority relabeling architecture [15] and the Rotating-Priority-Queues (RPQ) scheduler [11]. These schedulers use so-called *calendar queues* [2, 10] that periodically relabel FIFOs to increase the priorities of queued packets.[1] Relabeling the FIFOs provides an efficient approximation to sorting, where the granularity of the approximation is determined by

*This work is supported in part by the National Science Foundation under Grant Nos. NCR-9309224 and NCR-9624106.

[1]Note that similar approaches have been used to approximate both fixed-rate and Virtual Clock schedulers [7, 10].

the relabeling period. The RPQ scheduler is distinguished from the priority labeling architecture in that it can provide worst-case delay guarantees, but neither scheduler has an efficiency that always exceeds that of a *Static Priority* (SP) scheduler which does not change the priorities of queues.

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$^+$ can approximate the efficiency of the EDF scheduler with arbitrary precision. Similar to the priority relabeling architecture and RPQ, the RPQ$^+$ scheduler employs a set of prioritized FIFO queues whose priorities are periodically rotated. 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 a SP scheduler. So far no existing packet scheduler that tries to approximate EDF can satisfy all of the above characteristics. We derive necessary and sufficient conditions for schedulability in RPQ$^+$, that is, conditions for which all packets are guaranteed to be transmitted within their delay constraints. Using these conditions, we demonstrate that RPQ$^+$ is identical to SP if the FIFO queues are never rotated. 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 [8].

The remainder of this paper is structured as follows. We first discuss the framework of a bounded-delay service in Section 2. In Section 3 we present operations of the RPQ$^+$ scheduler and introduce an implementation approach for shared-memory switches. In Section 4 we derive necessary and sufficient conditions for schedulability in RPQ$^+$ and show that RPQ$^+$ is a hybrid between EDF and SP. We finally evaluate RPQ$^+$ in Section 5 using numerical examples as well as MPEG-compressed video traces.

2 Bounded-Delay Service Framework

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 [6]. 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 [4]:

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

In practice, traffic constraint functions for a connection are expressed with a small number of parameters using a *traffic model* [6, 12, 14, 18, 20]. As an example, we consider the traffic constraint function A^* for the (σ, ρ) traffic model [4] that corresponds with the traffic admitted by a leaky bucket mechanism [17]. The (σ, ρ) model uses a burstiness parameter σ and a rate parameter ρ, and its worst-case traffic is given as follows [4]:

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

2.2 Scheduling 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 [5], or controlling the distortion of the arrivals by reshaping the traffic to conform to A_i^* at each switch using so-called traffic shaping mechanisms [20].

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, SP, and RPQ packet schedulers that are derived in [12]. In the EDF condition we use s_j^{max} to denote the maximum transmission time of a packet from connection j. In the SP and RPQ conditions, 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, $d_p < d_q$ for $p < q$. The

Packet Scheduler	Schedulability Conditions
EDF	For all $t \geq d_1$: $\quad t \geq \sum_{i \in \mathcal{C}} A_i^*(t - d_i) + \max_{d_j > t} s_j^{max}$
SP	For all p and all $t \geq 0$, there exists a $0 \leq \tau \leq d_p - s^{min}$ such that: $$t + \tau \geq \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(t+\tau) + \sum_{i \in \mathcal{C}_p} A_i^*(t) - s^{min} + \max_{q>p} s_q^{max}$$
RPQ	For all $t \geq d_1$: $\quad t \geq \sum_{i \in \mathcal{C}_1} A_i^*(t - d_1) + \sum_{p=2}^{P} \sum_{i \in \mathcal{C}_p} A_i^*(t - d_p + \Delta) + \max_{d_q > t - \Delta} s_q^{max}$

Table 1: Necessary and sufficient schedulability conditions.

RPQ scheduler also uses a so-called rotation parameter Δ that determines the frequency of FIFO relabeling; all supported delay bounds d_p are multiples of Δ, that is, $d_p = n_p\Delta$ for some integer n_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 The RPQ$^+$ Scheduler

In this section we introduce the *Rotating-Priority-Queues*$^+$ (RPQ$^+$) scheduler that approximates the optimal EDF scheduler. 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. Here we first describe the design principles and operations of the RPQ$^+$ scheduler. We then illustrate its operations using a simple example. Finally, we describe an efficient implementation strategy for shared-memory architectures where FIFO queues are implemented as linked lists.

3.1 Properties of RPQ$^+$

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.

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.

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 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^+. If a packet arrival occurs at the same time as a queue rotation, we assume that the queue rotation is performed before the packet arrives.

3.2 Illustration of RPQ$^+$ Scheduling

The operations of an RPQ$^+$ scheduler are best illustrated by means of a simple example. Figure 1 depicts the states of an RPQ$^+$ scheduler over the course of a queue rotation. We first consider Figure 1(a) which shows a snapshot of the scheduler. In this example, the scheduler is assumed to support 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 with index p in the figure. 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. Packets are assumed to arrive from the left in Figure 1, 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 of the figure. If packets are in the scheduler as shown in Figure 1(a), the next packet selected will be the packet from connection set $\mathcal{C}_1$ since FIFO 1 is the highest-priority nonempty queue.

We next consider the queue rotation illustrated in the figure. Figure 1 shows, from left to right, (a) the state of the queues before a queue rotation, (b) the concatenation step of the queue rotation, and (c) the promotion step of the queue rotation. The concatenation step shown in Figure 1(b) 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. When concatenating FIFO p and FIFO p^+ for $1 \leq p < P$, all packets from FIFO p^+ are inserted at the tail of FIFO p. Figure 1(c) shows the promotion of packets from FIFO p to FIFO $(p-1)^+$ for $p = 1, 2, 3$. Note that three new FIFO p queues are included in Figure 1(c) for new arrivals during the next rotation interval.

Observe that we do not specify a location to which packets in FIFO 0^+ are moved during a queue ro-

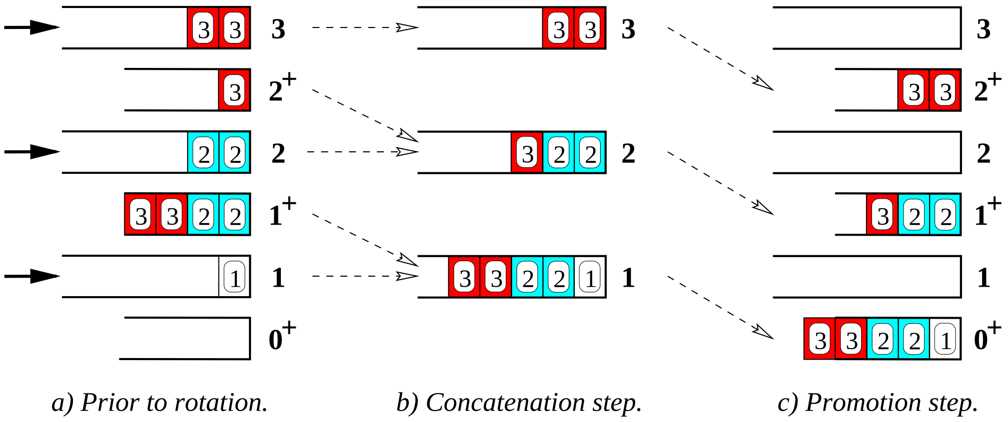


Figure 1: Example of RPQ$^+$ scheduling operations and queue rotations.

tation. 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 scenario, the delay bounds for the connection sets are selected such that FIFO 0^+ will necessarily be empty at the end of all rotation intervals. However, for services other than a bounded-delay service, RPQ$^+$ can be designed to either (1) discard all packets in FIFO 0^+ since they have necessarily violated their deadlines or (2) leave these packets in FIFO 0^+ and concatenate new packets to the end of the FIFO.

3.3 An Efficient Implementation of RPQ$^+$

The overhead for implementing RPQ$^+$ is identical to that of an SP scheduler except for the queue rotations. Here we investigate the operations required for implementing the RPQ$^+$ queue rotation in switches with shared-memory output buffering.

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 shared-memory buffers [1]. In switches with shared-memory output buffers, the FIFO queues of RPQ$^+$ can be implemented using linked lists, and queue rotations can be implemented with a small number of operations using simple pointer manipulations.

Figure 2 shows buffer management for an outgoing link of a shared-memory switch that employs the RPQ$^+$ scheduler. Figures 2(a) and 2(b) illustrate the state of the scheduler before and after a queue rotation, respectively. 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. Packets arriving to FIFO p are inserted at $T(p)$ its tail, and the scheduler selects packets for transmission from FIFO p at its head.Comparing Figure 2(b) with Figure 2(a), we notice that only a small number of pointer manipulations are needed to implement the queue rotation.

4 RPQ$^+$ Schedulability Conditions

In this section we present schedulability conditions for the RPQ$^+$ scheduler and show that its efficiency is between EDF and SP. We first present in Theorem 1 the exact, that is, necessary and sufficient, schedulability conditions for a general class of traffic constraint functions. We then show explicitly that RPQ$^+$ is superior to SP and that its efficiency increases monotonically for a certain class of rotation intervals, converging to EDF in the limit.

4.1 Exact Schedulability Conditions

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:*

$$t+\tau \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(\min\{t+\tau, t+d_p-d_q+\Delta\})+$$

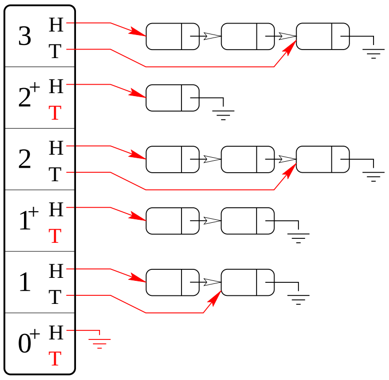
$$+\sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t+d_p-d_q) - s^{min} + \max_{r, d_r > t+d_p} s_r^{max} \quad (3)$$

A proof of the theorem is provided in [19].

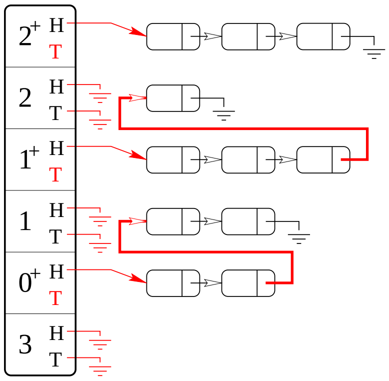
4.2 RPQ$^+$ as a Hybrid between EDF and SP

Here we show that the RPQ$^+$ scheduler is a hybrid between EDF and SP in the sense that (i) RPQ$^+$ always achieves an efficiency at least as good as SP, (ii) the efficiency of RPQ$^+$ is nondecreasing as the rotation interval Δ is reduced,and (iii) the efficiency of RPQ$^+$ approaches that of EDF as $\Delta \to 0$. We consider each of these properties in turn.

To show Property (i), we argue that any set of connections schedulable with SP is also schedulable with



(a) Immediately before rotation.



(b) After queue rotation.

Figure 2: Implementation of RPQ$^+$ queue rotation.

RPQ$^+$. Our argument relies on the following 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 for all priorities p and for all $t \geq 0$, there exists a τ with $\tau \leq d_p - s^{min}$ such that:*

$$t + \tau \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(t+\tau) + \sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t + d_p - d_q) - s^{min} + \max_{r, d_r > t+d_p} s_r^{max} \quad (4)$$

A complete proof of Lemma 1 is provided in [19]. With the lemma, note that the necessary condition for SP in equation (4) implies the sufficient condition for RPQ$^+$ in equation (3). This shows that RPQ$^+$ always achieves an efficiency at least as high as SP.

In order to show Property (ii), we observe that the right-hand-side of equation (3) increases with Δ. As required, the efficiency of RPQ$^+$ is nondecreasing as the rotation interval Δ is reduced.

Finally, to show Property (iii), we compare RPQ$^+$ with EDF using a sufficient (but not necessary) schedulability condition for RPQ$^+$ that has a formulation similar to the exact EDF condition in Table 1. We obtain this condition directly from Theorem 1 by substituting $\tau = d_p - s^{min}$ in equation (3):

Corollary 1 *A set $\mathcal{C}$ of connections given by $\{A_i^*, d_i\}_{i\in\mathcal{C}}$ is RPQ$^+$-schedulable with rotation interval Δ if for all priorities p and for all $t \geq d_1$ the following condition holds:*

$$t \geq \sum_{q=1}^{p-1} \sum_{i\in\mathcal{C}_q} A_i^*(t - d_q + \Delta) + \sum_{q=p}^{P} \sum_{i\in\mathcal{C}_q} A_i^*(t - d_q) + \max_{r, d_r > t} s_r^{max} \quad (5)$$

Comparing the condition in equation (5) with the EDF condition, we see that the only difference in the two conditions is the rotation interval Δ. These two conditions become identical in the limit as $\Delta \to 0$, verifying that an RPQ$^+$ scheduler can approximate EDF with arbitrary precision.

5 Evaluation

In this section we compare the efficiency of the RPQ$^+$ scheduler against schedulers EDF and SP 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$^+$ and the conditions from Table 1 for EDF and SP.

5.1 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 [12]. 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:

$$A^*(t) = \sigma + \rho t \quad (6)$$

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.

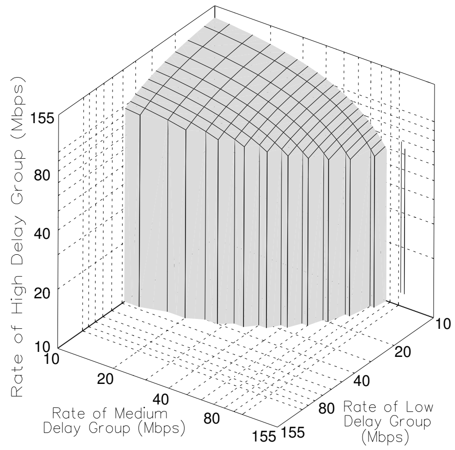
Delay Group	Index j	Delay Bound d_j (ms)	Burst σ_j (cells)	Rate ρ_j (Mbps)
Low	1	12	4000	10-155
Medium	2	24	2000	10-155
High	3	36	4000	10-155

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

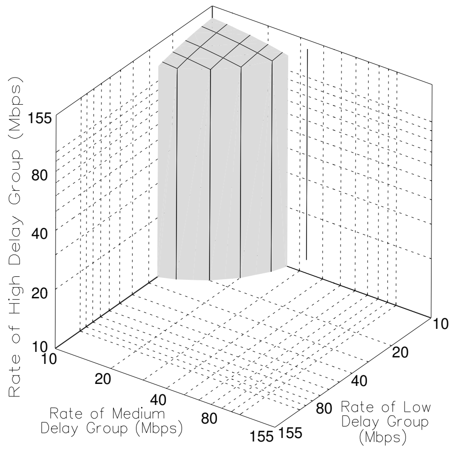
We present our results with two sets of graphs in Figures 3 and 4. The schedulable region for different schedulers are shown using three-dimensional surface plots. We show the transmission rates for which all connection groups are admissible; the volume beneath a curve includes all operating points for which all packets are guaranteed to be transmitted before their deadlines. Note that the axes in these figures have a logarithmic scale.

Figure 3 depicts the schedulable regions for EDF and SP schedulers. Since EDF is the optimal packet scheduler with respect to number of admissible connections, the region shown in Figure 3(a) 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 shown in Figure 3(b).

In Figure 4, we show schedulable regions of the RPQ$^+$ packet scheduler for feasible rotation intervals in the range $\Delta = 2 - 12$ms. For this example, the number of queues that must be maintained for a particular choice of Δ is given by $72/\Delta$. Note that for all choices of rotation interval Δ, the RPQ$^+$ schedulable region is superior to that of SP in Figure 3(b). 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 4(a)-(d) that the schedulable region increases as the rotation interval is decreased, approximating EDF closely when $\Delta = 2$ms.



(a) EDF Scheduler.



(b) SP Scheduler.

Figure 3: Benchmark schedulable regions.

5.2 MPEG Example

In this experiment all traffic characterizations are obtained from publicly-available traces of MPEG video [16]. 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 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 [3, 18]:

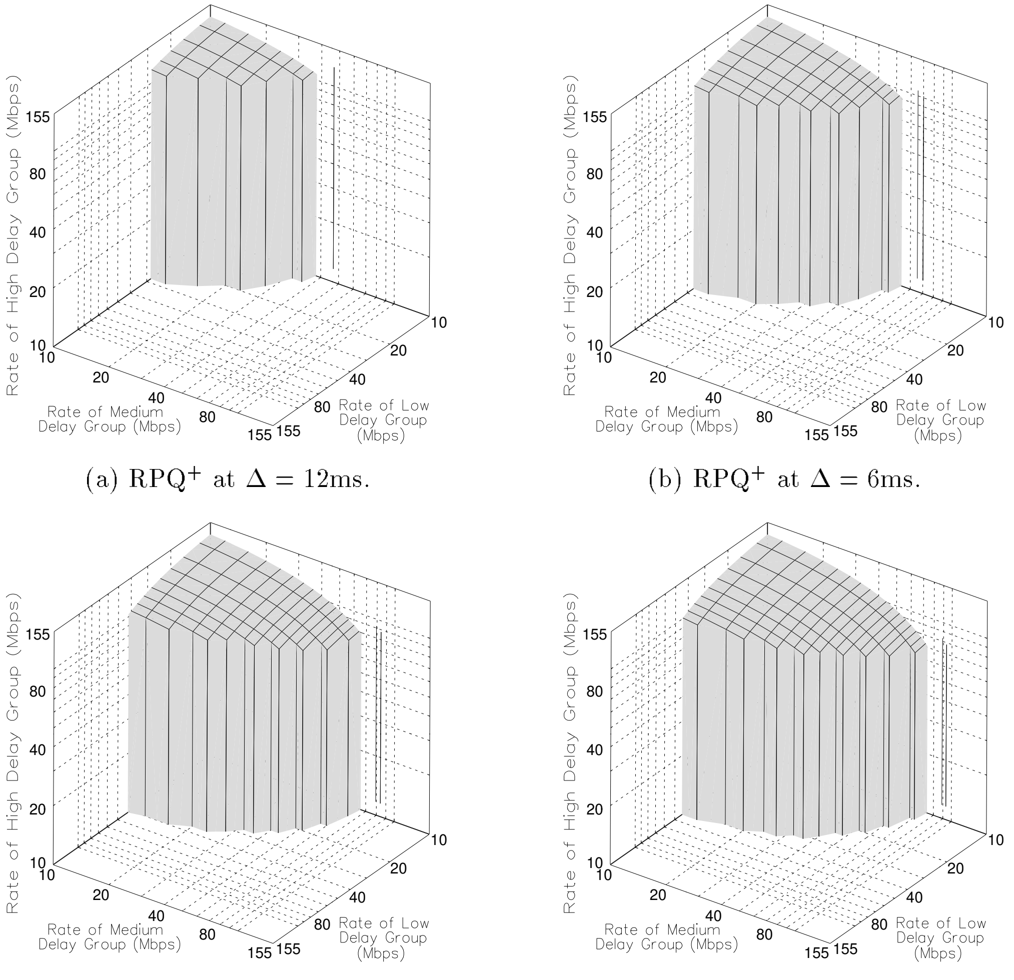
$$E^*(t) = \sup_{\tau \geq 0} A[\tau, \tau + t] \qquad \forall t \geq 0 \qquad (7)$$

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: $d_{Settop} = 100$ms and $d_{Bond} = 200$ms.

Figure 5 illustrates the number of connections that can be supported at their delay constraints for the EDF, SP, and RPQ$^+$ schedulers as well as for a peak-rate allocation scheme.[2] 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

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



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

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

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

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

Figure 4: Schedulable regions for RPQ$^+$.

lines for RPQ$^+$ at different values of rotation interval Δ. For the RPQ$^+$ curves we show the value of Δ as well as 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 is superior to SP when the number of higher-priority *Settop* connections is large. Note in Figure 5 that RPQ$^+$ is identical to SP for $\Delta = 100$ms, and smaller values of Δ result in higher efficiency. For $\Delta = 10$ms, a point of operation requiring 40 FIFOs, RPQ$^+$ closely approximates EDF.

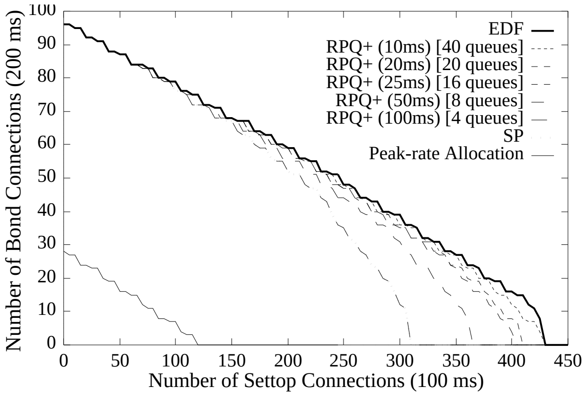


Figure 5: Example of RPQ$^+$ for MPEG video sequences.

6 Conclusions

In this paper we presented the novel Rotating-Priority-Queues$^+$ (RPQ$^+$) packet scheduler. We showed that in a bounded-delay service RPQ$^+$ can achieve a network utilization that approximates that of the optimal Earliest-Deadline-First (EDF) scheduler. We also 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] N. Berry. Asynchronous Transfer Mode (ATM) Switch Technology and Vendor Survey. Technical Report NAS-95-001, National Aeronautics and Space Administration, January 1995.

[2] 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.

[3] 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.

[4] 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.

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

[6] 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.

[7] N. R. Figueira. *A New Approach to the Control of Real-Time Traffic in Packet Switching Data Networks*. PhD thesis, University of California at San Diego, June 1995.

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

[9] 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.

[10] 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.

[11] 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.

[12] J. Liebeherr, D. E. Wrege, and Domenico Ferrari. Exact Admission Control in Networks with Bounded Delay Services. To appear: *IEEE/ACM Transactions on Networking*, December 1996.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

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

[18] 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. *IEEE/ACM Transactions on Networking*, 4(3):352–362, June 1996.

[19] D. E. Wrege and J. Liebeherr. A Near-Optimal Packet Scheduler for QoS Networks. Technical Report CS-96-12, University of Virginia, June 1996.

[20] H. Zhang. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE*, 83(10):1374–1399, October 1995.