Priority Queue Schedulers with Approximate Sorting in Output Buffered Switches

Jörg Liebeherr, Dallas E. Wrege

Abstract— **All recently proposed packet scheduling algorithms for output-buffered switches which support QoS transmit packets in some priority order, for example, according to deadlines, virtual finishing times, eligibility times, or other timestamps that are associated with a packet. Since maintaining a sorted priority queue introduces significant overhead, much emphasis of QoS scheduler design is put on methods to simplify the task of maintaining a priority queue. In this study, we consider an approach which attempts to approximate a sorted priority queue at an output buffered switch. The goal is to trade off less accurate sorting for lower computational overhead. Specifically, this paper presents a scheduler which approximates the sorted queue of an Earliest-Deadline-First (EDF) scheduler. The approximate scheduler is implemented using a set of prioritized FIFO queues which are periodically relabeled. The scheduler can be efficiently implemented with a fixed number of pointer manipulations, thus, enabling an implementation in hardware. Necessary and sufficient conditions for the worst-case delays of the scheduler with approximate sorting are presented. Numerical examples, including traces based on MPEG video, demonstrate that in realistic scenarios, scheduling with approximate sorting is a viable option.**

Keywords— **Packet Switching, Output Buffering, Multiplexing, Deterministic Service, Quality-of-Service.**

I. Introduction

One of the core components of a Quality-of-Service (QoS) network is the *packet scheduling* algorithm which determines the transmission order of packets at the output buffers of switches. In recent years, numerous packet schedulers have been proposed for QoS networks, and excellent surveys on the topic are available [2], [22], [42], [44].

Almost all packet scheduling algorithms for QoS networks which have been considered recently transmit packets in a priority order. For example, in fair queueing schedulers which approximate GPS [30], the priority order is determined by *virtual finishing times* [3], [17], [30], [36], *virtual start times* [18], or other derived timestamp values [35]. The Earliest-Deadline-First (EDF) scheduler transmits packets in the order of *deadlines* [8], [13], [15], [25]. In addition, *traffic shapers* [15], [16], [32], [43] which enforce that traffic entering a scheduler conforms to a given traffic specification, typically maintain a priority queue which sorts packets according to increasing *conformance times*, i.e., the times at which packets conform to their traffic specification. All of these schedulers have in common that they require an implementations of a sorted priority queue.

The computational overhead necessary to maintain a sorted priority queue is a potential bottleneck in high-speed packet switching. In general, the algorithmic complexity for maintaining a sorted priority queue with N arbitrary entries is $O(\log N)$ in the worst-case. To realize a sorted priority queue in a high-speed network one can exploit the parallelism feasible in a hardware implementation [7], [14], [19], [23], [28], [33], or use efficient data structures [5], [12], [21], [37], [40].

In this study we take the approach of reducing the complexity of a sorted priority queue by approximating the sorting operations, thus, trading off less accurate sorting for reduced computational complexity. As in [25], [26], [31], [32], we consider a scheduler which maintains a set of prioritized FIFO queues, and periodically modifies the priorities of the FIFO queues. The scheduler always transmits a packet from the highest-priority FIFO queue which contains a packet. In Figure 1 we show a scheduler with P FIFO queues; a lower index indicates a higher priority. Each FIFO queue is associated with a range of timestamp values; FIFO p is associated with timestamps in the range $[(p-1)\,\Delta, p\,\Delta]$. So, the total range of timestamps with P queues is given by $[0, P\,\Delta]$. Each packet that arrives to the scheduler is timestamped. Depending on the scheduler used, the timestamp can be a deadline, virtual finishing time, or other value. A newly arrived packet is inserted into *FIFO* p, if its timestamp is in $((p-1)\,\Delta, p\,\Delta]$.

J. Liebeherr is with the Computer Science Department, University of Virginia, Charlottesville, VA 22903. E-mail: jorg@cs.virginia.edu. D. E. Wrege is with IBM Corporation, Research Triangle Park, NC 27709, E-mail: wrege@raleigh.ibm.com. This work is supported in part by the National Science Foundation under Grant No. NCR-9624106 (CAREER) and ANI-9730103. Earlier results on the RPQ^+ scheduler have appeared in the Proceedings of IEEE Infocom '97.

Periodically, the labels of the FIFO queues are modified as follows: All queues FIFO p ($p > 1$) are relabeled as FIFO $(p-1)$, and FIFO 1 is relabeled as FIFO P. Thus, the FIFOs can be thought of as having performed a rotation as shown in Figure 2. The frequency of the queue rotations must be matched with the range of timestamp values assigned to each FIFO queue. In our example, a rotation should be performed once every Δ time units. Each relabeling of FIFO queues increases the priority level of a queue, and, with it, the priority level of the packets in the queue. A result of the relabeling is that packets get roughly sorted in the order of timestamps. Several recently proposed shaping and scheduling algorithms have employed this concept of rotating FIFO queues [24], [26], [31], [32].

Using rotating FIFO queues to implement an approximately sorted priority queue introduces two sources of inaccuracy. First, two packets that arrive to the same FIFO queue are stored in the order of their arrival, and not necessarily in the order of their timestamp values. Second, suppose a (tagged) packet with timestamp $p\Delta$ arrives shortly before a queue rotation. Upon arrival, this packet will be put in *FIFO* p, but the immediately following rotation will relabel this queue as *FIFO* $(p-1)$. Packets with timestamps as small as $(p-1)\ \Delta$, which arrive after the rotation to *FIFO* $(p-1)$, will be stored behind the tagged packet even though the tagged packet has a larger timestamp value.

Intuitively, the accuracy of approximating a priority queue with rotating FIFO queues improves as Δ is decreased. The key advantage of the scheduler is that insertion and deletion of packets have a complexity proportional to P, the number of FIFO queues. The complexity is independent of the total number of packets in the scheduler.

In this study we show how to approximate the sorted priority queue of an EDF scheduler. In EDF, each packet is timestamped with a *deadline* set equal to the sum of the packet arrival time and a per-session delay bound, and packets are transmitted in increasing order of deadlines [11], [15], [16], [25], [45]. We consider a *deterministic service*, that is, a service where all packets from a session satisfy worst-case end-to-end delay bounds [9], [11]. For such a service, EDF has been shown to have *optimal efficiency*[1], in that it, among all scheduling algorithms, supports the most sessions with delay guarantees [15], [25]. A scheduler that approximates EDF with rotating FIFO queues should satisfy the following properties:

(P1) Analytical schedulability conditions[2] should be available.

(P2) By decreasing the value of Δ, and appropriately increasing the number of FIFO queues, the efficiency of the scheduler should increase.

(P3) The efficiency of the scheduler should not be worse than a scheduler with prioritized FIFO queues that are never rotated.

Devising a packet scheduling algorithms that satisfies these properties proves to be difficult. The first proposals using rotating FIFO queues [26], [31] did not investigate schedulability conditions. In [25] we proposed a scheduler, called *Rotating Priority Queues (RPQ)*, that satisfies properties (P1) and (P2), but not property (P3). In this paper we present a scheduler, called *Rotating Priority Queues*$^+$ (RPQ^+), that satisfies all three properties listed above. An important result is that the number of FIFO queues needed for this scheduler is twice the number of priority levels.

The remainder of this paper is structured as follows. In Section II we provide some background on QoS networks with a bounded-delay service. In Section III we show that an overly simplistic design of rotating FIFO queues may result in poor performance of the scheduler. In Section IV we discuss the operations of the RPQ$^+$ scheduler and discuss an approach to implement the scheduler. In Section V we present necessary and sufficient conditions for schedulability in RPQ$^+$ and prove that it satisfies properties (P1) – (P3) from above. In Section VI we evaluate our scheduler using numerical examples as well as MPEG-compressed video traces. In Section VII we offer some conclusions.

II. Networks with a Deterministic Services

Consider a packet scheduler at the output port of some switch in the network. The set of sessions with traffic at this scheduler is denoted by $\mathcal{C}$. We assume that the sessions are partitioned into P priority classes, $\mathcal{C} = \bigcup_{1 \le p \le P} \mathcal{C}_p$, with $\mathcal{C}_p$ denoting the set of priority-p sessions. All sessions in $\mathcal{C}_p$ have the same delay bound d_p, and we assume $d_p < d_q$ whenever $p < q$. Thus, the priority index of a session is low if its delay bound is short. A bounded-delay service provides worst-case delay guarantees to all packets from a session. The delay bound d_p is a firm upper bound on the delay of any packet from a priority-p session. We use L_p^{max} to denote the maximum packet size in a priority-p session. L^{min} denotes the minimum packet size for all sessions.

[1]Within the context of guaranteeing QoS, the efficiency of a scheduler with a given transmission rate is measured in terms of the number of sessions for which QoS guarantees can be satisfied.

[2]Schedulability conditions for a given scheduler determine if QoS guarantees for a set of sessions can be satisfied.

For a given session $i \in \mathcal{C}$, let $A_i[\tau, \tau+t]$ denote the total session-i traffic which arrives to the scheduler in time interval $[\tau, \tau+t]$. The worst-case traffic entering the scheduler is expressed in terms of an envelope function A_i^* [10], [25] as follows:

$$A_i[\tau, \tau+t] \leq A_i^*(t) \qquad \forall t \geq 0, \forall \tau \geq 0 \quad (1)$$

We assume that envelope functions are subadditive, that is:

$$A_i^*(t_1) + A_i^*(t_2) \geq A_i^*(t_1+t_2) \qquad \forall t_1, t_2 \geq 0 \quad (2)$$

Most envelope functions used in practice are concave, piecewise linear functions, since it is relatively easy to device traffic policing mechanisms which enforce such envelopes [1], [4]. Here, the traffic on a session i is characterized by a set of parameters $\{\sigma_{ik}, \rho_{ik}\}_{k=1,2,\ldots,K}$ with envelope function

$$A_i^*(t) = \min_{k=1,2,\ldots,K} \{\sigma_{ik} + \rho_{ik} t\} \qquad \forall t \geq 0 \quad (3)$$

If an envelope function has only one linear segment ($K = 1$), the session traffic is referred to as *leaky-bucket constrained.* Since subadditivity is a weaker notion than concavity, all concave envelope functions are also subadditive.

For each session $i \in \mathcal{C}_p$, the local delay bound d_p at a switch accounts for the queueing and transmission delay at the output buffer. A packet on session i arriving to the output buffer at time t is assigned a deadline of $t + d_p$. Before a packet enters the scheduler, we assume that a shaping mechanism [16], [43] enforces that the packet is not in violation of its envelope function. The shaping mechanism holds a packet until it complies to the envelope of its session [16]. It has been proven that a shaping mechanism does not increase the maximum end-to-end delay [16], [41] of a packet. As a consequence, bounds for maximum end-to-end delays for single hop routes can be easily extended to multi-hop routes. Hence, in the remainder of the paper, we restrict our attention to the delay at a single output port of a switch with transmission rate R.

For a given scheduler, we say that a set $\mathcal{C}$ of sessions with envelope functions $\{A_i^*\}_{i \in \mathcal{C}}$ and delay bounds $\{d_p\}_{p=1,2,\ldots P}$, is *schedulable* if a deadline violation cannot occur for any session i which complies to its envelope A_i^*. The conditions which determine that a set of sessions is schedulable, so-called *schedulability conditions*, constitute the admission control test for a deterministic service. The efficiency of a packet scheduler with transmission rate R is measured by the number of sessions with given envelope functions that can be supported without deadline violations. We say that a scheduler X is *more efficient* than scheduler Y (given identical transmission rates), if X can support more sessions with delay guarantees than Y.

Next we state necessary and sufficient schedulability conditions for Earliest-Deadline-First (EDF) and Static Priority (SP) packet schedulers from [25]. These two schedulers serve as benchmarks for our work on approximating EDF. By comparing the efficiency of an approximate scheduler with that of EDF we are able to quantify how well we approximate the sorted priority queue of EDF. Since EDF yields, among all scheduling algorithms optimal efficiency for a deterministic service [15], [25], the efficiency attainable with EDF gives us an upper bound. An SP scheduler with P priority levels is implemented with a fixed number of P FIFO queues. (We assume that a lower priority index indicates a higher priority.) Arriving packets from priority p are placed into the p-th FIFO queue. The SP scheduler always selects the packet from the head of the highest-priority FIFO queue that contains a packet. The architecture of an SP scheduler is similar to the scheduler shown in Figure 1, but where FIFO queues are never relabeled. Since we expect that relabeling FIFO queues improves the efficiency of a scheduler, the efficiency of SP serves us as a lower bound for the efficiency we wish to attain. Note, however, that a scheduler which does not satisfy property (P3) from Section I may be less efficient than SP.

We assume a set of sessions $\mathcal{C}$ with P priority levels, where a session $i \in \mathcal{C}_p$ has an envelope function A_i^* and delay bound d_p. For leaky-bucket constrained envelopes the schedulability conditions simplify to a closed form (see Table 1 in Section VI).

Theorem 1: **(Liebeherr, Wrege, Ferrari [25])** A set of sessions $\mathcal{C}$ is schedulable in an EDF scheduler if and only if for all $t \geq d_1$:

$$R \cdot t \geq \sum_{p=1}^{P} \sum_{i \in \mathcal{C}_p} A_i^*(t - d_p) + \max_{d_q > t} L_q^{max} \quad (4)$$

Theorem 2: **(Liebeherr, Wrege, Ferrari [25])** A set of sessions $\mathcal{C}$ is schedulable in an SP scheduler if and only if for all p and all $t \geq 0$, there exists a $0 \leq \tau \leq d_p - L^{min}$ such that:

$$\begin{aligned} R \cdot (t+\tau) \quad \geq \quad & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(t+\tau) + \\ & + \sum_{i \in \mathcal{C}_p} A_i^*(t) - L^{min} + \max_{q > p} L_q^{max} \quad (5) \end{aligned}$$

III. Rotating FIFO Queues and Rotation Anomaly

Consider the scheduler for sorting FIFO queues shown in Figure 1 with P FIFO queues and assume local delay bounds $d_p = p\Delta$ $(1 \leq p \leq P)$. When a priority-p packet arrives to the scheduler it is added to *FIFO* p. Every Δ time units the queues are rotated as discussed in Section I. *FIFO* p is relabeled as *FIFO* $(p-1)$ for $p > 1$, and *FIFO* 1 is relabeled as *FIFO* P.

There is one problem with this design. Namely, if a packet resides in the highest priority queue, *FIFO* 1, at the time of queue rotation it will be in the lowest priority queue, *FIFO* P, after the rotation. We can avoid this problem by adding a queue *FIFO* 0, give this queue highest priority, and never insert a newly arriving packet directly into *FIFO* 0. Then, during a queue rotation, *FIFO* p is relabeled as *FIFO* $(p-1)$ for $p \geq 1$ and *FIFO* 0 is relabeled as *FIFO* P. Now, whenever *FIFO* 0 contains a packet at the time of a rotation, this packet has a deadline violation. Thus, ensuring that *FIFO* 0 is always empty at the time of a queue rotation is a necessary schedulability condition. This architecture was proposed and analyzed in [25] and is called *Rotating-Priority-Queues* (RPQ) scheduler.

In Figure 3 we illustrate the operations of an RPQ scheduler with three priority levels where $d_1 = \Delta$, $d_2 = 2\Delta$, and $d_3 = 3\Delta$. Figure 3(a) shows a snapshot of the FIFO queues at some time $0 \leq t < \Delta$. Packets are labeled with their priority index. Figure 3(b) depicts the content of the queues after a rotation at time Δ. Figure 3(c) shows a snapshot in time interval $\Delta \leq t < 2\Delta$, and Figure 3(d) shows the result of a second queue rotation.

In RPQ, the overhead for performing the rotation can be as low as incrementing a single counter. The counter is incremented by one after each queue rotation. Then, the current *FIFO* p is obtained simply by computing "(counter $+ p$) mod $(P+1)$".

Given a set of sessions $\mathcal{C}$ with P priority levels, where a session $i \in \mathcal{C}_p$ has an envelope function A_i^* and delay bound $d_p = \Delta p$, the necessary and sufficient schedulability conditions for the RPQ scheduler as derived in [25] are as follows:

Theorem 3: **(Liebeherr, Wrege, Ferrari [25])** A set of sessions $\mathcal{C}$ is schedulable in an RPQ scheduler with parameter Δ if and only if for all $t \geq d_1$:

$$R \cdot t \geq \sum_{i \in \mathcal{C}_1} A_i^*(t - d_1) + \sum_{p=2}^{P} \sum_{i \in \mathcal{C}_p} A_i^*(t - d_p + \Delta) + \max_{d_q > t - \Delta} L_q^{max} \quad (6)$$

A comparison of the condition for RPQ in Equation (6) with the EDF condition in Theorem 1 shows that RPQ approximates EDF arbitrarily closely if Δ is selected sufficiently small.

If queues are never rotated, that is $\Delta = \infty$, RPQ reduces to an SP scheduler. One would expect that any feasible selection of $\Delta < \infty$ will yield a higher achievable utilization than SP. However, often this is not the case. Decreasing Δ of an RPQ scheduler may not improve its efficiency. In fact, one can devise examples where EDF and SP accept the same number of sessions, but RPQ admits less sessions for any finite choice of Δ. We refer to the problem that reducing the length of the rotation interval in RPQ does not increase its efficiency as *rotation anomaly.* A consequence of the *rotation anomaly* is that RPQ may be less efficient than an SP scheduler. In Section VI we will present examples which illustrate the effects of the rotation anomaly.

Figure 4 presents an attempt to illustrate the cause of rotation anomaly in RPQ. In Figure 4(a) we have created an arrival scenario for an RPQ scheduler with 3 priority levels. We assume that time T is a time instant of a queue rotation. Suppose that shortly before the rotation, at time $T - \varepsilon$, 3 packets from priority 2 arrive to an empty scheduler. Further, suppose that shortly after the rotation, at time $T + \varepsilon$, on packet from priority 1 arrives to the scheduler. Figures 4(b) and 4(c) depict the content of the RPQ scheduler at times $T - \varepsilon$ and $T + \varepsilon$, respectively. Due to the rotation at time T, the priority-2 packets will be transmitted before the priority-1 packet. Since $T - \varepsilon + 2\Delta > T + \varepsilon + \Delta$, the RPQ scheduler will not transmit these packets in the order of their deadlines. Hence, in this situation, RPQ does not transmit packets in the same order as an EDF scheduler. Note that, in the same arrival scenario, an SP scheduler would transmit packets in the same order as an EDF scheduler.

The rotating anomaly in RPQ lets us pose the following question: *Is it feasible to devise a scheduler with rotating FIFO queues which approximates the sorted priority queue of an EDF scheduler, yet, without rotating anomaly?* This question will be answered in the next section.

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

We next present a scheduler for approximating EDF with rotating FIFO queues which does not have the rotation anomaly. We will prove that the scheduler, which we will call *Rotating-Priority-Queues*$^+$ (RPQ$^+$), has always better efficiency than an SP scheduler.

The principal idea is to add newly arriving packets in intermediate FIFO queues which are located between the FIFO queues of RPQ. With these intermediate queues, RPQ$^+$ needs twice as many FIFO queues as RPQ, for any fixed selection of Δ. However, the increase of the achievable efficiency of RPQ$^+$ outweighs its cost. In Section VI we will see that RPQ$^+$ with rotation interval Δ typically outperforms RPQ with rotation interval $\Delta/2$.

A. RPQ$^+$ Scheduling

An RPQ$^+$ scheduler employs $2P$ ordered FIFO queues, which are indexed, from highest to lowest priority: $0^+, 1, 1^+, 2, 2^+, \ldots, (P-1), (P-1)^+, P$. The scheduler always selects a packet from the highest-priority nonempty FIFO queue for transmission. All new packets arriving on a session in group $\mathcal{C}_p$ are placed in *FIFO* p. Arriving packets are never placed directly into *FIFO* p^+ for any p.

Example: Figure 5 depicts an RPQ$^+$ scheduler which supports three session groups $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, with delay bounds $d_p = p\Delta$ for $p = 1, 2, 3$. An RPQ$^+$ scheduler which supports these session groups requires 6 FIFO queues with indices $\{0^+, 1, 1^+, 2, 2^+, 3\}$. In the scheduler shown in Figure 5 the next packet selected will be the packet at the head of queue 0^+.

The FIFO queues for an RPQ$^+$ scheduler are relabeled (*rotated*) every Δ time units. An RPQ$^+$ queue rotation is a two-step process. The first step is called *concatenation step* and the second is called *promotion step*. In the concatenation step, the current *FIFO* p and *FIFO* p^+ are merged to form *FIFO* p ($1 \le p < P$). In this step, all packets from *FIFO* p^+ are concatenated to the end of *FIFO* p. In the promotion step, *FIFO* p is relabeled as *FIFO* $(p-1)^+$ ($1 \le p \le P$). At the end of the promotion step, new *FIFO* p queues (for $1 \le p \le P$) are created for new packet arrivals during the next rotation interval. Note that, after the promotion step, all packets reside in some *FIFO* p^+. If a packet arrival occurs at the same time as a queue rotation, the queue rotation is performed before the packet arrives.

Example: Figure 6 illustrates queue rotations and scheduling operations of an RPQ$^+$ scheduler over the course of three rotation intervals. Assuming that the scheduler begins operation at time 0, Figure 6(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. Figure 6(b)(i) depicts the state of the queues at time 2Δ. (Between Figure 6(a)(iii) and Figure 6(b)(i), the packet in *FIFO* 0^+ and one of the packets from *FIFO* 1^+ have been transmitted. In addition, there were new arrivals to to all *FIFO*s p queues for $p = 1, 2, 3$.) In $[\Delta, 2\Delta)$, packet arrivals from session set $\mathcal{C}_p$ are placed into *FIFO* p, but packets from the same session group that arrived during the previous rotation interval reside in *FIFO* $(p-1)^+$. The second queue rotation at time 2Δ is illustrated in Figures 6(b)(ii) and 6(b)(iii). (Between Figure 6(b)(iii) and Figure 6(c)(i), all packets in *FIFO* 0^+ and one of the packets from *FIFO* 1^+ have been transmitted, and there were new arrivals to all *FIFO*s p for $p = 1, 2, 3$.) Figure 6(c)(i) depicts the RPQ$^+$ scheduler at time $3\Delta^-$, and Figures 6(c)(ii) and 6(c)(iii) illustrate the two phases of the queue rotation at time 3Δ. Note in Figure 6(c)(iii) that packets from all 3 session groups have moved to the highest-priority *FIFO* 0^+ at time 3Δ.

B. An Implementation of RPQ$^+$

We briefly discuss how the queue rotation in RPQ$^+$ can be implemented.

The overhead for implementing RPQ$^+$ reduces to that of an SP scheduler except for the queue rotations. If output buffer memory consists of a single shared memory pool on a per-port basis [14], the rotating FIFO queues of RPQ$^+$ can be efficiently implemented with a small number of pointer manipulations.

In Figure 7 we show that the number of pointer manipulations can be kept small. In Figure 7(a) we show the pointers necessary to maintain an RPQ$^+$ scheduler. Here, all packets queued in the scheduler are implemented as a global linked list. The pointer HP is a global pointer to the head of the list which identifies the next packet to be transmitted. For each FIFO queue, there is a pointer, denoted by $T(\cdot)$, which identifies the last packet in the queue. For example, $T(1^+)$ points to the last packet that is stored in *FIFO* 2^+. Finally, $T(0^+)$ indicates an empty pointer. Note that Figure 7(a) reflects the state of the queues in Figure 6(c)(i). The implementation of the queue rotation shown in Figures 7(a)–(c) corresponds to the queue rotation in Figure 6(c)(i)–(iii).

The concatenation step, shown in Figure 7(b), is performed by setting $T(p) = T(p^+)$ for all $1 \le p < 3$, and by resetting all $T(p^+)$ pointers. The results of the promotion step are shown in Figure 7(c). Each queue *FIFO* p is renamed as *FIFO* $(p-1)^+$ for $p > 0$, each queue *FIFO* p^+ is renamed *FIFO* p, and the queue *FIFO* 0^+ is relabeled as *FIFO* 3.

V. Admission Control for RPQ$^+$

In this section we discuss the schedulability conditions of RPQ$^+$ and characterize properties of the scheduler. Theorem 4 presents the exact, that is, necessary and sufficient, schedulability conditions for RPQ$^+$ for the general class of subadditive envelope functions.

Theorem 4: A set of sessions $\mathcal{C}$ with P priority classes and with delay bounds $d_p = \Delta p$ is RPQ$^+$-schedulable with parameter Δ if and only if for all priorities p and for all $t \geq 0$ there exists a τ with $0 \leq \tau \leq d_p - L^{min}$ such that:

$$\begin{aligned} R \cdot (t+\tau) \geq & \\ & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(\min\{t+\tau, t+d_p-d_q+\Delta\}) + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} + A_i^*(t+d_p-d_q) - \\ & - L^{min} + \max_{r, d_r > t+d_p} L_r^{max} \end{aligned} \tag{7}$$

A complete proof of the theorem is presented in Appendix A.

Using the schedulability conditions, we establish a set of important properties of the RPQ$^+$ scheduler: (1) the efficiency of RPQ$^+$ is non-decreasing as the rotation interval Δ is decreased, (2) the efficiency of RPQ$^+$ approaches that of EDF as $\Delta \to 0$, and (3) RPQ$^+$ always achieves at least the efficiency of SP, that is, RPQ$^+$ does not exhibit the rotation anomaly. Together with Theorem 4, we thus have constructed a scheduler that satisfies the properties stated in Section I: (P1) is satisfied since Theorem 4 gives us exact schedulability conditions, (P2) is satisfied via items (1) and (2) below, and (P3) is satisfied via item (3).

(1) **The efficiency of RPQ$^+$ is non-decreasing as Δ is reduced.**
This claim is easily seen by observing that the right-hand side of Equation (7) increases with Δ. (Recall that all delay bounds are of the form $d_p = p\Delta$ for all p, that is, delay bounds are multiples of Δ.)

(2) **The efficiency of RPQ$^+$ approaches that of EDF as $\Delta \to 0$.**
In order to compare RPQ$^+$ with EDF, we present a sufficient schedulability condition for RPQ$^+$ that has a formulation similar to the exact EDF conditions from Theorem 1. We obtain the sufficient condition directly from Theorem 4 by substituting $\tau = d_p - L^{min}$ in Equation (7).

Corollary 1: A set of sessions $\mathcal{C}$ with P priority classes with delay bounds $d_p = \Delta p$ is RPQ$^+$-schedulable with parameter Δ if for all priorities p and for all $t \geq d_p$:

$$\begin{aligned} R \cdot t \quad \geq \quad & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(t-d_q+\Delta) + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i^*(t-d_q) + \max_{r, d_r > t} L_r^{max} \end{aligned} \tag{8}$$

Comparing the condition in Equation (8) with the EDF condition in Equation 4, we see that the only difference in the two conditions is the rotation interval Δ. The two conditions become identical in the limit as $\Delta \to 0$.

(3) **RPQ$^+$ is free of the rotation anomaly.**
To show that RPQ$^+$ is never worse than SP, we prove that any set of connections which is schedulable with SP is also schedulable with RPQ$^+$. Our argument relies on a necessary condition for SP schedulability.

Lemma 1: If a set $\mathcal{C}$ of connections schedulable in SP, then for all priorities p and for all $t \geq 0$ there exists a τ with $\tau \leq d_p - L^{min}$ such that:

$$\begin{aligned} R \cdot (t+\tau) \quad \geq \quad & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(t+\tau) + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i^*(t+d_p-d_q) - \\ & - L^{min} + \max_{r, d_r > t+d_p} L_r^{max} \end{aligned} \tag{9}$$

A proof of the lemma is given in Appendix B. With Lemma 1, we have that schedulability in SP implies Equation (9). By inspection of Equation (7) in Theorem 4, we can see that Equation (9) is a sufficient schedulability condition for RPQ$^+$. Therefore, schedulability in SP implies schedulability in RPQ$^+$, or, equivalently, RPQ$^+$ always accepts at least the same number of connections as SP.

In Figure 8 we show the same arrival scenario as in Figure 4. We illustrate how RPQ$^+$ manages to transmit packets in the order of deadlines in such a scenario. Figure 8(a) depicts the same arrival scenario as in Figure 4(a). Figure 4(b) shows the scheduler at time $T - \varepsilon$ after the arrival of the priority-2 packets. Since the priority-2 packets will be rotated to the *FIFO* 1^+ at time T (see Figure 4(c)), the priority-1 packet which arrives at time $T+\varepsilon$ will be transmitted before the priority-2 packets. As a result, the RPQ$^+$ scheduler transmits these packets in the order of their deadlines.

VI. Evaluation

In this section we compare the efficiency of the RPQ$^+$ scheduler against EDF, SP, and RPQ in two sets of experiments. In the experiments, we want to answer questions on the performance of the respective schedulers, such as:

- How many FIFO queues do schedulers with rotating FIFO queues, such as RPQ and RPQ$^+$, require to closely approximate the sorted priority queue of an EDF scheduler?
- Does the *rotation anomaly* in RPQ manifest itself as an observable phenomenon in a high-speed network environment?

For network traffic, we consider leaky-bucket constrained traffic in the first experiment, and traces of MPEG-compressed video in the second experiment. In both experiments we use the most accurate, i.e., necessary and sufficient, admission control tests from Theorems 1–4 for each of the respective packet schedulers.

A. Numerical Example

In the first experiment we compute the *schedulable region* [20] of the packet schedulers for a set of three session groups, i.e., we vary the traffic rate of each session group and plot the rates for which delay bounds can be guaranteed.

We consider sessions at the scheduler of an output port with transmission rate 155 Mbps. The three session groups have leaky-bucket constrained traffic with envelope function $A_j^* = \sigma_j + \rho_j t$ for session group j. For leaky-bucket constrained sources, the schedulability conditions from Theorems 1 – 4 can be greatly simplified. In Table I we show the expressions for the schedulability conditions for EDF, SP, RPQ. Note that, with leaky-bucket constrained traffic, all conditions simplify to closed form expressions.

Table II shows the traffic and QoS parameters for the session groups. For a session group with index j, the table shows the delay bound d_j at the scheduler, as well as the burst size σ_j, measured in 53-byte ATM cells, and the range of rates ρ_j, varied between 10 and 155 Mbps.

In Figures 9 – 11 we illustrate the schedulable regions for the different schedulers. The volume beneath each curve contains the operating points for which the transmission rates of the session groups are schedulable. The axes in these figures have a logarithmic scale.

Figure 9(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 must not exceed the rate of the transmission link, that is, $\sum_{j=1}^{3} \rho_j < 155$ Mbps.

In Figures 9(b) and 9(c), we depict the schedulable regions for EDF and SP packet schedulers, respectively. Since EDF is the optimal packet scheduler with respect to the number of schedulable sessions, the region shown in Figure 9(b) will contain the region of any other packet scheduler. Observe that the schedulable region for EDF is much larger than that for SP as shown in Figure 9(c). Without rotation anomaly, the regions for EDF and SP serve as upper and lower bounds, respectively, for the schedulable regions of our rotating FIFO queue schedulers.

In Figure 10, we illustrate schedulable regions of the RPQ scheduler for feasible values of Δ 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 $1 + (d_3/\Delta)$, meaning that RPQ needs between 4 and 37 queues. In Figures 10(a)-(f) note that the schedulable region increases as Δ, the time between rotations, is decreased. The region for RPQ approaches that of EDF quickly. For $\Delta = 1$ms in Figure 10(f), the region is close to that of EDF. Comparing the regions in Figures 10(a) and 10(b) with the region for SP in Figure 9(c), we see the effects of the rotation anomaly (see Section III) where RPQ is inferior to SP.

In Figure 11 we show the schedulable regions of the RPQ$^+$ scheduler for ranges $\Delta = 1 - 12$ms. The number of required queues for an RPQ$^+$ scheduler with rotation interval Δ in this example is $2d_3/\Delta = 72/\Delta$. Comparing the results for RPQ$^+$ regions with the benchmark regions from Figure 9, we note that for all choices of Δ, the RPQ$^+$ schedulable region is superior to that of SP in Figure 9(c). Even for $\Delta = 12$ms, the largest possible choice of Δ for this example, the RPQ$^+$ schedulable region completely contains the SP region. Figures 11(a)–(f) also show that the schedulable region increases as Δ is decreased, closely approximating EDF when $\Delta = 1$ms.

When comparing the schedulability regions for RPQ$^+$ in Figure 11 with those for RPQ in Figure 10, observe that, for all choices of Δ, the RPQ$^+$ scheduler achieves a larger schedulability region than the corresponding RPQ scheduler.

Figure 12 is an attempt to condense the results for the EDF, SP, RPQ, and RPQ$^+$ schedulers in a single graph. For a packet scheduler Σ, we let $V^{\Sigma}(\Delta)$ denote the volume of the schedulable region in Figures 9–11 as a function of Δ. Since EDF and SP are not dependent on Δ, both $V^{SP}(\Delta)$ and $V^{EDF}(\Delta)$ are constant as Δ is varied. Letting V^{∞} denote the volume of the schedulable region shown in Figure 9(a), we define the ratio of $V^{\Sigma}(\Delta)$ and V^{∞} expressed as a percentage:

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

In Figure 12 we plot the resulting values for a packet scheduler as a function of Δ. For example, the value of about 15% for RPQ with $\Delta = 4$ms means that the volume contained in the schedulable region of RPQ with $\Delta = 4$ms (from Figure 10) is 15% of the volume of the region in Figure 9(a).

In Figure 12 we can make several noteworthy observations. First, for values of $\Delta > 4$ms, the RPQ scheduler performs worse than SP, clearly demonstrating that rotation anomaly is a hazard in RPQ scheduling. Second, for identical values of Δ, RPQ$^+$ clearly outperforms RPQ. Note that for the same value of the rotation interval, RPQ$^+$ requires twice as many FIFO queues as RPQ. However, even when comparing the results for RPQ with rotation interval Δ to that of RPQ$^+$ with rotation interval $\Delta/2$, that is, when comparing RPQ and RPQ$^+$ on the basis that the schedulers use the same total number of FIFO queues, RPQ$^+$ turns out to be superior most of the time.

B. MPEG Example

In this experiment we use two MPEG video traces as traffic sources: a thirty-minute segment of the entertainment movie *Goldfinger* (*"Bond"*) and 200 seconds of a video conference recorded using a set top camera ("*Settop*") [34]. Both traces were encoded in software at 24 frames/second with frame size 384×288 and frame pattern IBBPBBPBBPBB [27].

We again consider a packet scheduler that operates at a 155 Mbps output port, 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* [6], [38] to characterize the traffic a video sequence, where the empirical envelope E^* of a video sequence with traffic A is given by:

$$E^*(t) = \sup_{\tau \geq 0} A[\tau, \tau + t] \qquad \forall t \geq 0 \tag{10}$$

The empirical envelope of a traffic trace specifies the tightest envelope function satisfying Equation (1) for this trace. In other words, for the same scheduler, empirical envelopes will admit more sessions than any other envelope function.

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

Figure 13 illustrates the number of sessions that can be supported at their delay constraints for the EDF, SP, RPQ, and RPQ$^+$ schedulers as well as for a peak-rate allocation scheme.[3] For each packet scheduler, we plot the maximum number of admissible *Bond* sessions as a function of the number of *Settop* sessions. For example, all packet schedulers (except the peak-rate scheme) can support 96 *Bond* sessions if there are no *Settop* sessions at the switch, and EDF can simultaneously support 60 *Bond* and 200 *Settop* sessions.

We observe in Figure 12 that all packet schedulers admit more sessions than a peak-rate allocation. Additionally, EDF is superior to SP when the number of higher-priority *Settop* sessions is large. We observe in Figure 13(a) that RPQ is inferior to SP when the number of high-priority sessions is small, demonstrating again that RPQ is vulnerable to the rotation anomaly. Note in Figure 13(b) that RPQ$^+$ is identical to SP for $\Delta = 100$ms, and smaller values of Δ result in higher efficiency.

VII. Conclusions

We investigated approximations of sorted priority queues in packet schedulers of output buffered packet switches. Since the computational overhead for maintaining a sorted priority queue is a potential bottleneck for high-speed packet switching, an approximation which trades off less accurate sorting for lower computational overhead is attractive for packet switching at high transmission rates. We considered approximations of the sorted queue of Earliest-Deadline-First (EDF) scheduling. The approximation was implemented using a set of prioritized FIFO queues which are periodically relabeled. We derived admission control tests for the approximate schedulers, and compared the efficiency of the approximation with existing scheduling algorithms. We pointed to a severe problem, called *rotation anomaly*, that may arise when the approximation uses an insufficient number of FIFO queues. We avoided rotation anomalies by equipping the scheduler with two FIFO queues for each priority level.

Since most QoS schedulers, in particular, all fair queueing algorithms, require a sorted priority queue, the approach of approximating the sorted queue with rotating FIFOs can be extended to those schedulers. For such work, our notion of *rotation anomaly* and the avoidance of the anomaly using additional FIFO queues offers some guidance.

As part of an ongoing effort, we have devised a

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

VHDL specification of the RPQ$^+$ scheduler [29]. Using Synchronous Static RAM with 10 ns access times and a clock rate of 100 Mhz, we can perform the queue rotation of 64 FIFO queues in 3 ns. In this implementation, all pointers $TP(\cdot)$ and HP (see Section 4.2) were kept in on-chip registers.

References

[1] ATM Forum, ATM Forum Traffic Management Specification Version 4.0, April 1996.

[2] C.M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne. Real-Time Communication in Packet-Switched Networks. *Proceedings of the IEEE*, 82(1):122–139, January 1994.

[3] J. C. R. Bennett and H. Zhang. WF2Q: Worst-case Fair Weighted Fair Queueing. In *Proc. IEEE Infocom '96*, March 1996.

[4] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. IETF RFC 1633, July 1994.

[5] 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.

[6] 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.

[7] H. J. Chao. A Novel Architecture for Queue Management in the ATM Network. *IEEE Journal on Selected Areas In Communications*, 9(7):1110–1118, September 1991.

[8] F. M. Chiussi and V. Sivaraman. Achieving High Utilization in Guaranteed Services Networks using Early-Deadline-First Scheduling. In *Proc. 6th IEEE International Workshop on Quality of Service (IWQoS'98)*, pages 209–217, May 1998.

[9] D. D. Clark, S. Shenker, and L. Zhang. Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms. In *Proc. ACM Sigcomm*, pages 14–26, August 1992.

[10] 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.

[11] 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.

[12] 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.

[13] V. Firoiu, D. Towsley, and J. Kurose. Efficient Admission Control for Piecewise Linear Traffic Envelopes at EDF Schedulers. *IEEE/ACM Transactions on Networking*, 6(5):558–570, October 1998.

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

[15] 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.

[16] 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.

[17] S. J. Golestani. A Self-Clocked Fair Queueing Scheme for Broadband Applications. In *Proc. IEEE Infocom '94*, pages 636–646, June 1994.

[18] P. Goyal, H. M. Vin, and H. Cheng. Start-Time Fair Queueing: a Scheduling Algorithm for Intergrated Services Packet Switching Networks. In *Proc. ACM Sigcomm*, pages 157–168, 1996.

[19] M. R. Hashemi and A. Leon-Garcia. A General Purpose Cell Sequencer/Scheduler for ATM Switches. In *Proc. IEEE Infocom '97*, April 1997.

[20] 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.

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

[22] S. Keshav. *An Engineering Approach to Computer Networking.* Addison Wesley, 1997.

[23] P. Lavoie and Y. Savaria. A Systoclic Architecture for Fast Stack Sequential Decoders. *IEEE Transactions on Communications*, 42(2/3/4):324–334, February-April 1994.

[24] 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.

[25] J. Liebeherr, D. E. Wrege, and Domenico Ferrari. Exact Admission Control in Networks with Bounded Delay Services. *IEEE/ACM Transactions on Networking*, 4(6):885–901, December 1996.

[26] 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.

[27] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall (Eds.). *MPEG Video Compression Standard.* Chapman and Hall, 1997.

[28] S.-W. Moon, J. Rexford, , and K. Shin. Scalable Hardware Priority Queue Architectures for High-Speed Packet Switches. In *Proc. Real-Time Applications Symposium*, pages 203–212, June 1997.

[29] Y. Ou. A VLSI Implementation of RPQ$^+$ Scheduling. Master's Project, Department of Electrical Engineering, Polytechnic University, May 1998.

[30] A. K. Parekh and R. G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Single-Node Case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[31] 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.

[32] J. Rexford, F. Bonomi, A. Greenberg, and A. Wong. Scalable Architectures for Integrated Traffic Shaping and Link Scheduling in High-speed ATM Switches. *IEEE Journal on Selected Areas In Communications*, 15(6):938–950, June 1997.

[33] J. W. Roberts, R. E. Boyer, and M. J. Servel. A Real-Time Sorter with Applicatoin to ATM Traffic Control. In *Proc. ISS '95*, pages 278–282, 1995.

[34] 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.

[35] S. Shreedhar and G. Varghese. Efficient Fair Queueing Using Deficit Round Robin. *IEEE Transanctions on Networking*, 4(3):375–385, June 1996.

[36] D. Stiliadis and A. Varma. Design and Analysis of Frame-Based Fair Queueing: A New Traffic Scheduling Algorithm for Packet-Switched Networks. In *Proc. ACM Sigmetrics'96*, pages 104–115, May 1996.

[37] S. Suri, G. Varghese, and G. Chandranmenon. Leap Forward Virtual Clock. In *Proc. IEEE Infocom '97*, April 1997.

[38] 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.
[39] D.E. Wrege and J. Liebeherr. A near-optimal packet scheduler for QoS networks. In Proc. IEEE Infocom '97 Conference, April 1997.
[40] H. Zhang. *Service Disciplines for Packet-Switching Integrated-Services Networks.* PhD thesis, University of California - Berkeley, November 1993.
[41] H. Zhang. Providing End-to-End Performance Guarantees Using Non-Work-Conserving Disciplines, October 1995.
[42] H. Zhang. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE*, 83(10):1374–1399, October 1995.
[43] H. Zhang and D. Ferrari. Rate-Controlled Static-Priority Queueing. In *Proc. IEEE Infocom '93*, pages 227–236, April 1993.
[44] H. Zhang and S. Keshav. Comparison of Rate-Based Service Disciplines. In *Proc. ACM Sigcomm*, pages 113–121, Zurich, Switzerland, September 1991.
[45] Q. Z. Zheng and K. G. Shin. On the Ability of Establishing Real-Time Channels in Point-to-Point Packet Switching Networks. *IEEE Transactions on Communications*, 42(3):1096–1105, March 1994.

Appendix

I. Proof of Theorem 4

We first derive an expression for the workload transmitted before an arbitrary packet in RPQ$^+$ for packet arrivals with subadditive envelope functions described in Section II. This expression is central to proving the schedulability conditions. To simplify notation we assume without loss of generality that the transmission rate of the scheduler is normalized, that is, $R = 1$.

A. Workload Transmitted before an Arbitrary Packet

Assume that a (tagged) packet from a priority-p session $j \in \mathcal{C}_p$ arrives to an RPQ$^+$ scheduler at time t. Without loss of generality we assume that the scheduler is empty at time 0. We further assume that the packet is fully transmitted at time $t + \delta$. The tagged packet arrives after a queue rotation that occurred at time $t - \tau_\Delta$, where $0 \le \tau_\Delta < \Delta$. Queue rotations occur every Δ time units, and so we can express queue rotation times in terms of τ_Δ as follows:

$$\{(t - \tau_\Delta) + i\,\Delta \mid i \text{ an integer}\} \tag{11}$$

We will derive an expression for the total transmission time of all traffic in the scheduler at time $t + \tau$ that must be transmitted before the tagged priority-p packet with arrival time t can depart. This expression will be denoted as $W^{p,t}(t+\tau)$. We assume that the transmission time of the tagged packet is given by L, where $L^{min} \le L \le L_p^{max}$. Since the tagged packet completes transmission at time $t + \delta$, the packet begins transmission at time $t + \delta - L$. At the time the packet begins transmission, the value of $W^{p,t}$ is equal to L, that is, it includes only the tagged packet itself. Therefore, the departure time $t + \delta$ of the tagged packet satisfies:

$$\delta = L + \min\{z \ge 0 \mid W^{p,t}(t+z) = L\} \tag{12}$$

We next determine, for each priority level q, the arrivals from an arbitrary session q that are transmitted before the tagged packet. Let us first assume that the transmission of a packet can be preempted at any time by a packet arrival with higher precedence. We need to consider three types of arrivals: arrivals from sessions with the same priority level as the tagged packet ($q = p$), arrivals from sessions with a higher priority ($q < p$), and arrivals from sessions with a lower priority ($q > p$). We refer to Figure 14 for an illustration of all three cases.

(a) $\mathbf{q} = \mathbf{p}$: In RPQ$^+$, all packets from the same priority level $\mathcal{C}_p$ are transmitted in FIFO order. Therefore all packets from sessions in $\mathcal{C}_p$ that arrive in the

time interval $I_p = [0, t]$, are transmitted before the tagged packet.
(b) $\mathbf{q} < \mathbf{p}$: For a session from priority level $q < p$, the packets transmitted before the tagged packet are those that arrive before the tagged packet is rotated into *FIFO* $(q-1)^+$. The tagged packet will be in *FIFO* $(p-1)^+$ after the first queue rotation after t, i.e., at time $(t-\tau_\Delta)+\Delta$, and it will be in *FIFO* $(p-2)^+$ at time $(t-\tau_\Delta)+2\Delta$. More generally, at time $(t-\tau_\Delta)+(n+1)\Delta$, the tagged packet will have a higher priority than new arrivals from session set $\mathcal{C}_{p-n}$ for $n \geq 1$. Therefore, the time interval when arrivals from session $j \in \mathcal{C}_q$ are transmitted before the tagged packet is given by $I_q = [0, \min\{t+\delta, (t-\tau_\Delta)+(p-q+1)\Delta\}]$. The interval takes into account that the packet begins transmission at time $t+\delta-L$.
(c) $\mathbf{q} > \mathbf{p}$: For a session from a lower priority level $q > p$, all packets which, at time t, are in a queue *FIFO* r^+ $(r < p)$ will be transmitted before the tagged packet. Consider for example packets from session 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. More generally, for sessions in $\mathcal{C}_q$ with $q > p$, packets arriving in the time interval $I_q = [0, (t-\tau_\Delta)+(p-q)\Delta]$ will be transmitted before the tagged packet.

The intervals I_q describe the traffic that is transmitted before the tagged packet. However, these intervals assume that the transmission of a packet can be interrupted and preempted. To relax this assumption, let $t-\hat{\tau}$ be the last time before t when the RPQ$^+$ scheduler does not contain a packet that will be transmitted before the tagged packet. According to our previous considerations, $t-\hat{\tau}$ is the last time at which the scheduler does not have a backlog from packets in $\mathcal{C}_q$ $(q \leq p)$ that arrived in $[0, t-\hat{\tau}]$, nor does it have a backlog from packets in $\mathcal{C}_q$ $(q > p)$ that arrive in $[0, \min\{t-\hat{\tau}, (t-\tau_\Delta)+(p-q)\Delta]$. If we use $B_i(\tau)$ to denote the backlog in the RPQ$^+$ scheduler from session $i \in \mathcal{C}$ at time τ, $\hat{\tau}$ is given by:

$$\begin{aligned}\hat{\tau} = \min\{z \geq 0 \,|\, & \sum_{q=1}^{p} \sum_{i \in \mathcal{C}_q} B_i(t-z) + \\ & + \sum_{q=p+1}^{P} \sum_{i \in \mathcal{C}_q} B_i(\min\{t-z, (t-\tau_\Delta) + \\ & \qquad +(p-q)\Delta\}) = 0\} \qquad (13)\end{aligned}$$

Per construction, $W^{p,t}(t-\hat{\tau}) = 0$ and $W^{p,t}(x) > L$ in the entire interval $t-\hat{\tau} < x < t+\delta-L$. Now consider the packet from priority level q which is in transmission at time time $t-\hat{\tau}$ and has a remaining transmission time of $R(t-\hat{\tau})$. Since the transmission of this packet cannot be preempted, this packet will delay the departure time of all higher priority packets, including that of our tagged packet, by the amount $R(t-\hat{\tau})$.

We are now in a position to explicitly write an expression for $W^{p,t}(t+\tau)$. For all τ, $0 \leq \tau \leq \delta$ the expression is given by:

$$\begin{aligned}W^{p,t}(t+\tau) = & \sum_{i \in \mathcal{C}_p} A_i[t-\hat{\tau}, t] + \\ & + \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i[t-\hat{\tau}, \min\{t+\tau, (t-\tau_\Delta) + \\ & \qquad +(p-q+1)\Delta\}] + \\ & + \sum_{q=p+1}^{P} \sum_{i \in \mathcal{C}_q} A_i[t-\hat{\tau}, (t-\tau_\Delta)+(p-q)\Delta] + \\ & + R(t-\hat{\tau}) - (\hat{\tau}+\tau) \qquad (14)\end{aligned}$$

The first three terms on the right-hand-side of Equation (14) account for the arrival intervals I_q as derived previously, while the term $R(t-\hat{\tau})$ is the remaining transmission time of the packet that is in transmission 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.

Using the expression in Equation (14) and the condition in Equation (12) we will now prove sufficiency and necessity of the conditions in Theorem 4.

B. Sufficiency of Theorem 4

We show that the RPQ$^+$ scheduler transmits all packets before their deadlines if the inequality in Equation (7) holds for all times $t \geq 0$. We consider our tagged packet from session $j \in \mathcal{C}_p$ with size L $(L^{min} \leq L \leq L_p^{max})$ that arrives to the scheduler at time t and has a deadline of $t+d_p$. To show that the packet departs before its deadline, it is sufficient to find some time $\overline{\tau}$ with $\overline{\tau} \leq t+d_p-L$ such that $W^{p,t}(t+\overline{\tau}) \leq L$.

We start with the workload $W^{p,t}$ transmitted before the tagged packet as given in Equation (14). Consider $R(t-\hat{\tau})$, the remaining transmission time of a packet in transmission at time $t-\hat{\tau}$. Such a packet from a session $k \in \mathcal{C}_r$ has arrival time $t-t_0$, where $t_0 > \hat{\tau}$. By choice of $t-\hat{\tau}$, the time $t-t_0$ is restricted to:

$$t - t_0 > (t-\tau_\Delta) + d_p - d_r \qquad (15)$$

Using $t_0 > \hat{\tau}$, we can modify Equation (15) as follows:

$$d_r > \hat{\tau} + d_p - \tau_\Delta \tag{16}$$

The workload $W^{p,t}$ is maximal when $\tau_\Delta = 0$, and, since the maximum transmission time for a packet from a session in class $\mathcal{C}_r$ is given by L_r^{max}, we find the following upper bound on $R(t - \hat{\tau})$:

$$R(t - \hat{\tau}) \leq \max_{r; d_r > \hat{\tau} + d_p} L_r^{max} \tag{17}$$

Using the inequality in Equation (17) and setting $\tau_\Delta = 0$, we obtain the following bound on the workload $W^{p,t}$ from Equation (14):

$$\begin{aligned} W^{p,t}(t+\tau) \leq & \\ & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i[t - \hat{\tau}, \min\{t+\tau, t+(p-q+1)\Delta\}] + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i[t - \hat{\tau}, t + (p-q)\Delta] + \\ & + \max_{r; d_r > \hat{\tau} + d_p} L_r^{max} - (\hat{\tau} + \tau) \end{aligned} \tag{18}$$

Note that we were able to rewrite the second and third terms from Equation (14) as a single term in Equation (18). We can further bound the terms in the workload expression using the following two inequalities which use the properties of the envelope function A_i^*:

$$\begin{aligned} & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i[t - \hat{\tau}, \min\{t+\tau, t+(p-q+1)\ \}] \leq \\ & \qquad \leq \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(\min\{\hat{\tau}+\tau, \hat{\tau}+(p-q+1)\Delta\}) \end{aligned} \tag{19}$$

$$\begin{aligned} & \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i[t - \hat{\tau}, t + (p-q)\Delta] \leq \\ & \qquad \leq \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i^*(\hat{\tau} + (p-q)\Delta) \end{aligned} \tag{20}$$

Combining the inequalities in Equations (19) and (20) with the workload expression in Equation (18), we obtain:

$$\begin{aligned} W^{p,t}(t+\tau) \leq & \\ & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(\min\{\hat{\tau}+\tau, \hat{\tau}+(p-q+1)\Delta\}) + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i^*(\hat{\tau} + (p-q)\Delta) + \\ & \quad + \max_{r; d_r > \hat{\tau} + d_p} L_r^{max} - (\hat{\tau} + \tau) \end{aligned} \tag{21}$$

Since the packet size L is such that $L \geq L^{min}$, the condition in Equation (21) implies that the tagged packet is in transmission at time $t + \overline{\tau}$ and will complete transmission at time $t + \overline{\tau} + L^{min}$ Since $\overline{\tau} + L^{min} \leq d_p$, the tagged packet will be fully transmitted at time $t + d_p$ as required. □

C. Necessity of Theorem 4

Assume that there is a violation of the condition in Equation (7). That is, for some session set $\mathcal{C}_p$ and some time $\hat{t} \geq 0$, the following inequality holds for all τ with $0 \leq \tau \leq d_p - L^{min}$:

$$\begin{aligned} \hat{t} + \tau \quad < \quad & \sum_{q=1}^{p-1} \sum_{i \in \mathcal{C}_q} A_i^*(\min\{\hat{t}+\tau, \hat{t}+d_p-d_q+\Delta\}) + \\ & + \sum_{q=p}^{P} \sum_{i \in \mathcal{C}_q} A_i^*(\hat{t} + d_p - d_q) - \\ & - L^{min} + \max_{r, d_r > \hat{t} + d_p} L_r^{max} \end{aligned} \tag{22}$$

To show necessity of Theorem 4, we construct a feasible sequence of arrivals in which some packet will have a deadline violation.

We consider a scenario in which the RPQ$^+$ scheduler is empty up until time 0^-.[4] Assume that a packet of maximal size arrives to the scheduler at time 0^- from a session k in a session set $\mathcal{C}_r$ where $d_r > \hat{t} + d_p$. Such a packet has a size of $\max_{r; d_r > \hat{t} + d_p} L_r^{max}$. Further assume that all sessions in sets $\mathcal{C}_q$ with $d_q \leq \hat{t} + d_p$ submit as much traffic as possible starting at time 0, i.e., all sessions i submit $A_i^*(t)$ until time t, with one exception: For some session j with $j \in \mathcal{C}_p$, we delay the arrival of an amount of traffic of size L^{min} that would arrive before time $\hat{t}$ such that it arrives at time $\hat{t}$. Formally, if the last packet arrival from session $j \in \mathcal{C}_p$ before time $\hat{t}$ occurs at time $\hat{t} - z$ where:

$$z = \min\{z' \geq 0 \mid A_j^*(\hat{t} - z') < A_j^*(\hat{t})\}, \tag{23}$$

then a packet with length L^{min} is split off from this packet and delayed until time $\hat{t}$. This packet will be the *tagged packet*. Note that such a packet can be constructed if the packet size for all sessions $i \in \mathcal{C}_q$ is either constant or is such that $L^{min} \leq L^{max}/2$.

We also assume without loss of generality that a queue rotation occurs at time $\hat{t}$, i.e., $\tau_\Delta = 0$. Based on the above scenario, the workload $W^{p,\hat{t}}(\hat{t} + \tau)$ to be transmitted before the tagged packet at time $\hat{t} + \tau$

[4] 0^- denotes a time instant immediately before time 0.

is determined according to Equation (14) by:

$$W^{p,\hat{t}}(\hat{t}+\tau) = \sum_{q=1}^{p-1}\sum_{i\in\mathcal{C}_q} A_i^*(\min\{\hat{t}+\tau, \hat{t}+d_p-d_q+\Delta\}) + + \sum_{q=p}^{P}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+d_p-d_q) + + \max_{r;d_r>\hat{t}+d_p} L_r^{max} - \hat{t} \qquad (24)$$

Now, combining Equation (24) with our assumption in Equation (22), we find that, for all τ with $0 \le \tau \le d_p - L^{min}$:

$$W^{p,\hat{t}}(\hat{t}+\tau) > L^{min} \qquad (25)$$

Thus, the tagged packet with length L^{min} will not begin transmission before time $\hat{t}+d_p-L^{min}$ and, therefore, has a deadline violation. □

II. Proof of Lemma 1

To prove necessity of the condition in Lemma 1, we assume that Equation (9) is violated at some time $\hat{t}$, that is, there exists a priority p such that for all $\tau \le d_p - L^{min}$:

$$\hat{t}+\tau < \sum_{q=1}^{p-1}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+\tau) + + \sum_{q=p}^{P}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+d_p-d_q) - - L^{min} + \max_{r,d_r>\hat{t}+d_p} L_r^{max} \qquad (26)$$

We will show that a packet from some session $j \in \mathcal{C}_u$ with $u \ge p$ at the SP scheduler will have a deadline violation at or before time $t+d_p$.

We assume without loss of generality that arrivals to the SP scheduler are as follows. The scheduler is empty before time 0^-, and at time 0^- a packet arrives to the scheduler from session $k \in \mathcal{C}_r$, where $d_r > \hat{t}+d_p$, and the packet requires maximal transmission time. The packet size of such a packet is given by $\max_{r,d_r>\hat{t}+d_p} L_r^{max}$. Beginning at time 0 all sessions in $\mathcal{C}_q$ with $d_q \le \hat{t}+d_p$ submit a maximal amount of traffic to the scheduler according to A_i^*, with the exception that for all session sets $\mathcal{C}_q$, $q \ge p$, an amount of traffic equal to a packet with length L^{min} arriving before or at time $\hat{t}+d_p-d_q$ is delayed until time $\hat{t}+d_p-d_q$. In this construction there is a packet from each lower-priority session set $\mathcal{C}_q$ which has a deadline at time $\hat{t}+d_p$ and has minimal size. We refer to these packets collectively as the *delayed packets*. Among these packets, we consider the one that is transmitted last, and call it the *tagged packet*. This packet is from some session $j \in \mathcal{C}_u$ with $u \ge p$ and arrived to the scheduler at time $\hat{t}+d_p-d_u$. Note that the tagged packet begins transmission after time $\hat{t}$ since, by our construction, there is a delayed packet in $\mathcal{C}_p$ with arrival time $\hat{t}$. Assuming that the tagged packet begins transmission at time $\hat{t}+\delta-L^{min}$, the workload $W^{u,\hat{t}+d_p-d_u}(\hat{t}+\tau)$ transmitted before the tagged packet in time interval $[\hat{t}, \hat{t}+\delta-L^{min}]$ by an SP scheduler includes the following:

(a) By construction, since the tagged packet is transmitted after all other delayed packets, workload from session sets $\mathcal{C}_q$ $(q \ge p)$ arriving up until time $\hat{t}+d_p-d_q$ are transmitted before the tagged packet, i.e., the traffic $\sum_{q=p}^{P}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+d_p-d_q)$.

(b) Given that an SP scheduler always transmits the waiting packet with the highest priority, and also given that $u \ge p$, all traffic from session sets $\mathcal{C}_q$ $(q < p)$ in the scheduler at time $\hat{t}+\tau$ will be transmitted before the tagged packet. Thus, the traffic $\sum_{q=1}^{p-1}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+\tau)$ will be transmitted before the tagged packet.

(c) The low-priority packet arriving at time 0^- with size $\max_{r,d_r>\hat{t}+d_p} L_r^{max}$ is the only packet in the scheduler at time 0^-, and so it will be transmitted before the tagged packet.

Thus, the workload to be transmitted before the tagged packet with deadline $\hat{t}+d_p$ can be bounded as follows:

$$W^{u,\hat{t}+d_p-d_u}(\hat{t}+\tau) \ge \sum_{q=1}^{p-1}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+\tau) + + \sum_{q=p}^{P}\sum_{i\in\mathcal{C}_q} A_i^*(\hat{t}+d_p-d_q) + + \max_{r,d_r>\hat{t}+d_p} L_r^{max} - (\hat{t}+\tau) \qquad (27)$$

Combining Equation (27) with our assumption in Equation (26), we find that $W^{u,\hat{t}+d_p-d_u}(\hat{t}+\tau) > L^{min}$ for all $0 \le \tau \le d_p - L^{min}$, and thus the tagged packet will not be transmitted before its deadline. □

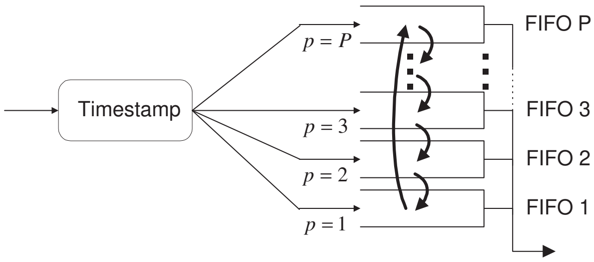
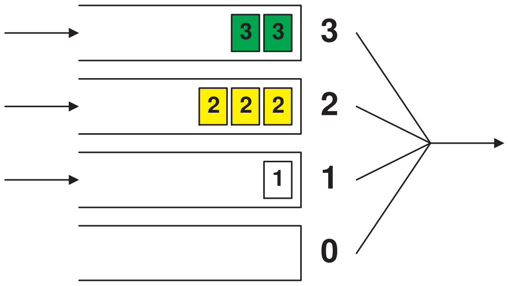
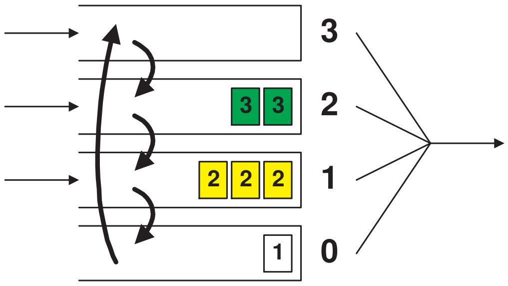


Fig. 1. Approximate Sorting with FIFO queues.

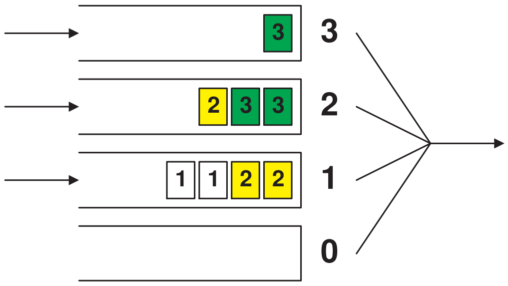
Fig. 2. Relabeling of FIFO Queues.



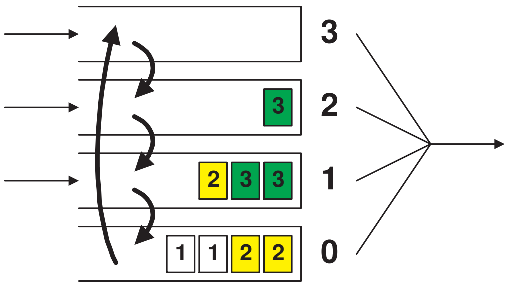
(a) $0 \leq t < \Delta$.



(b) $t = \Delta$.



(c) $\Delta \leq t < 2\Delta$.



(d) $t = 2\Delta$.

Fig. 3. RPQ Scheduler (with $P = 3$).

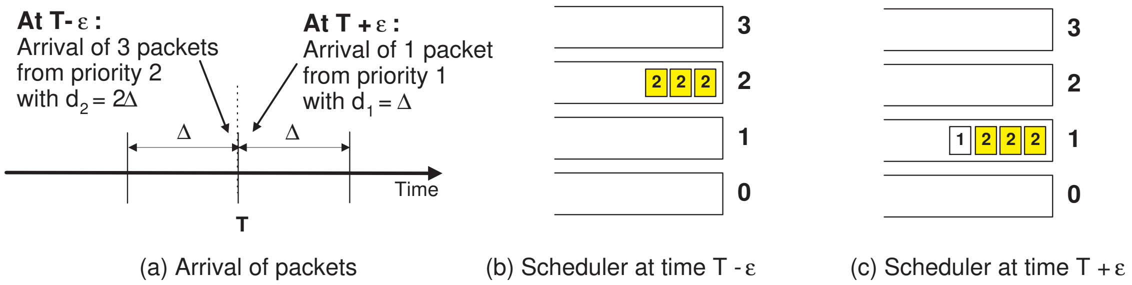


Fig. 4. Rotation Anomaly in RPQ.

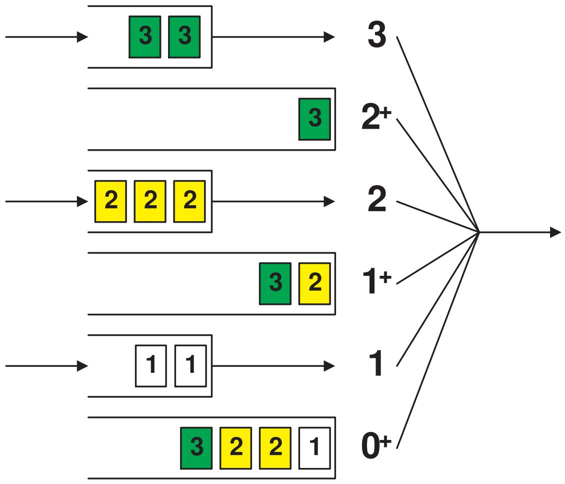
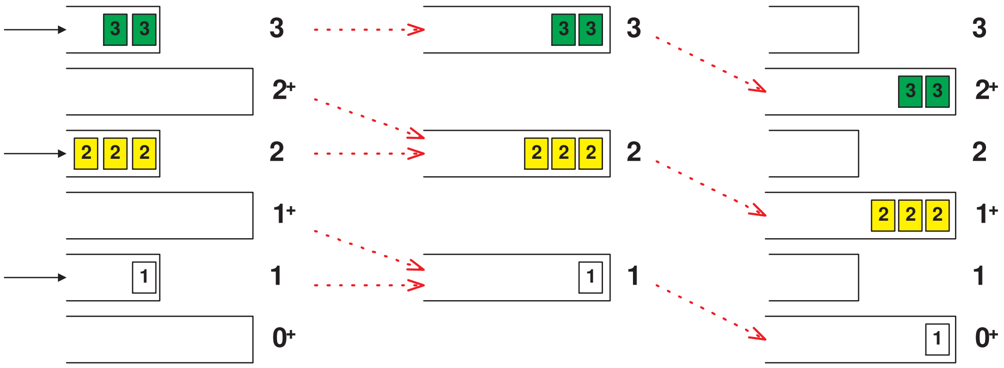
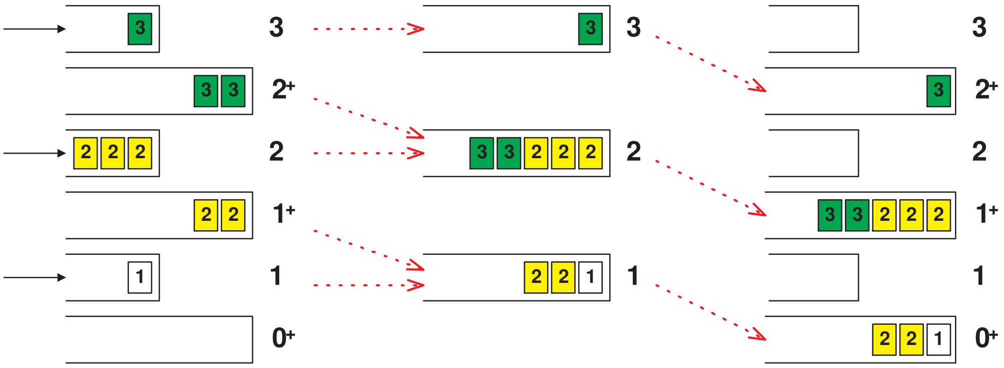


Fig. 5. RPQ$^+$ Scheduler (with $P = 3$).



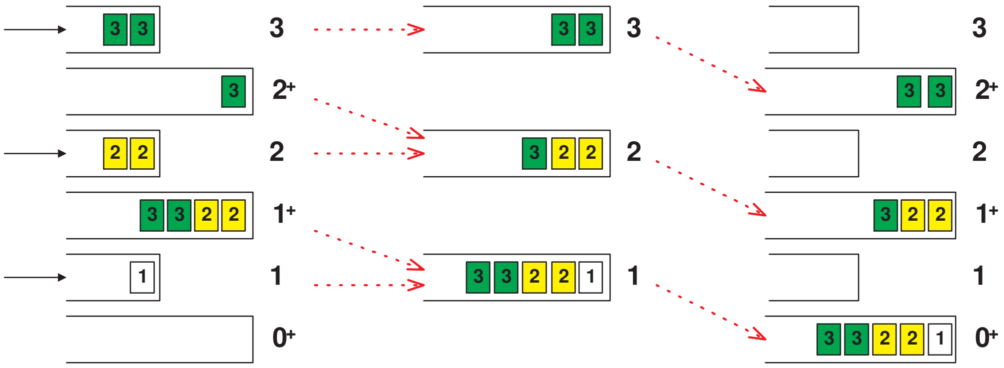
(i) Prior to Rotation *(ii) Concatenation Step* *(iii) Promotion Step*

(a) Rotation at time Δ.



(i) Prior to Rotation *(ii) Concatenation Step* *(iii) Promotion Step*

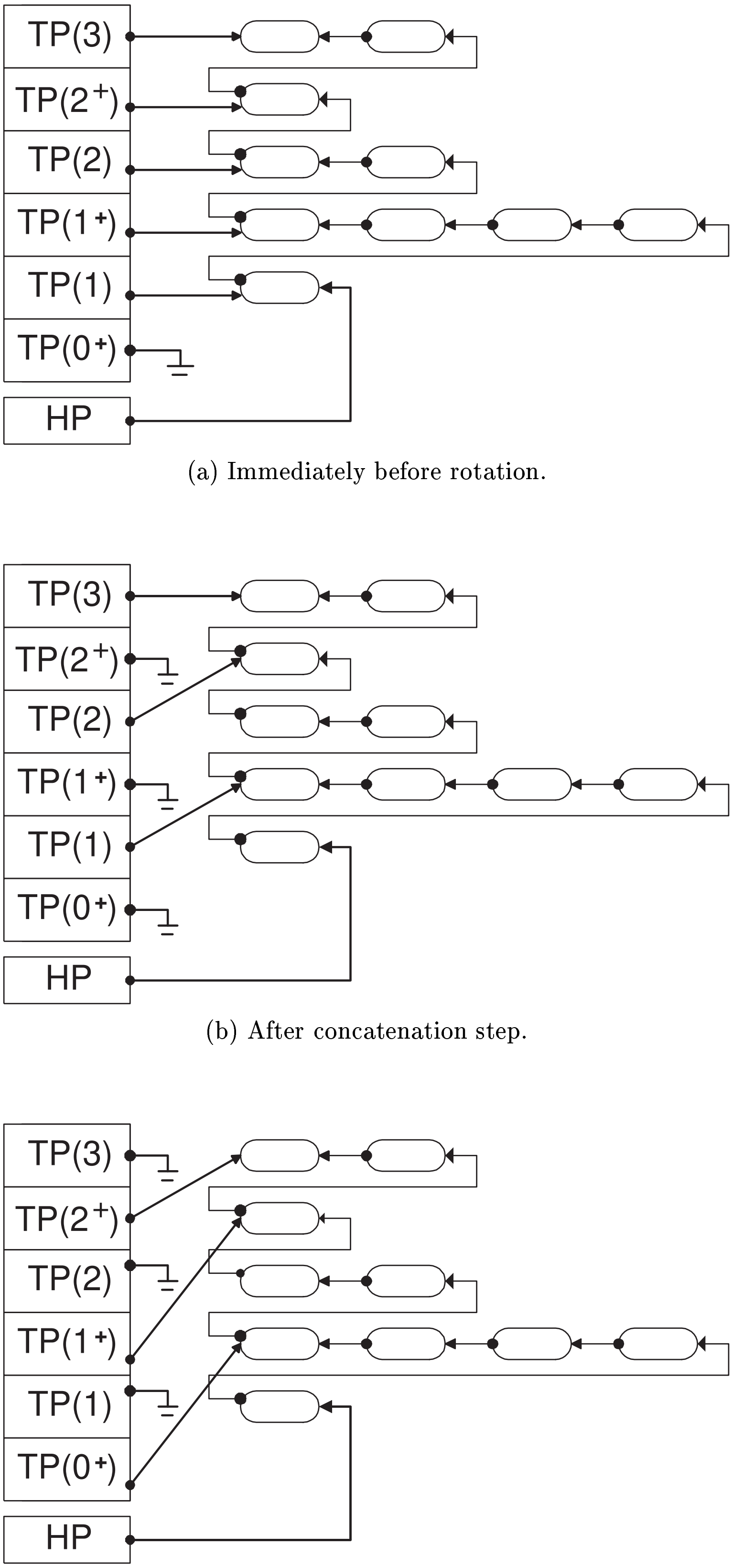
(b) Rotation at time 2Δ.



(i) Prior to Rotation *(ii) Concatenation Step* *(iii) Promotion Step*

(c) Rotation at time 3Δ.

Fig. 6. Example of RPQ$^+$ Scheduling Operations and Queue Rotations.



(a) Immediately before rotation.

(b) After concatenation step.

(c) After promotion step.

Fig. 7. Implementation of RPQ$^+$ Queue Rotation.

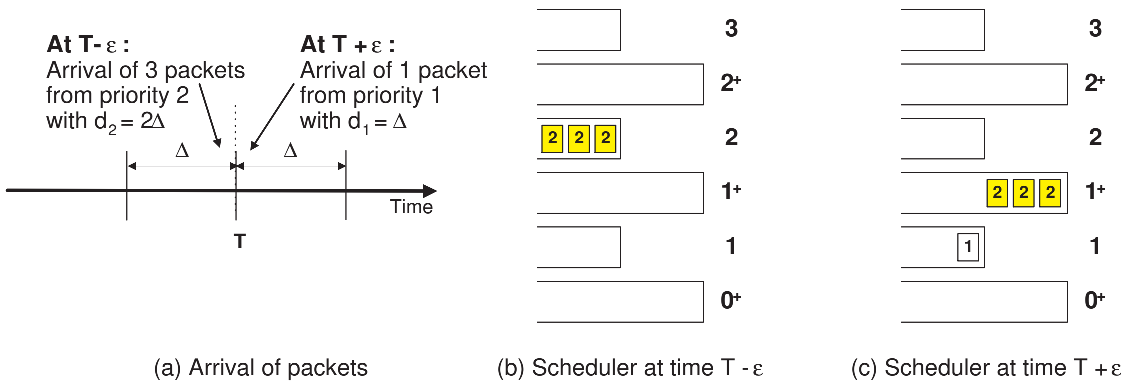
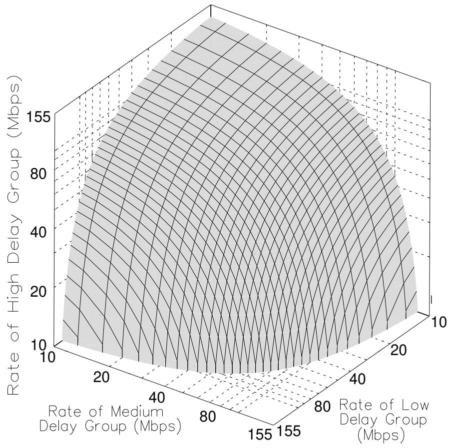
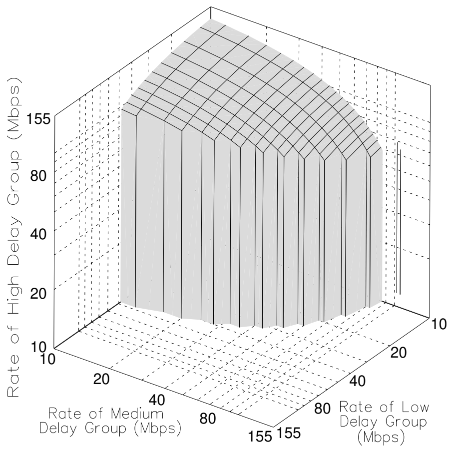


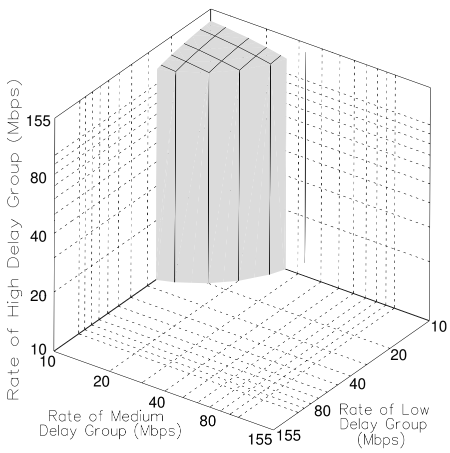
Fig. 8. No Rotation Anomaly in RPQ$^+$.



(a) Schedulable Region without Delay Constraints.

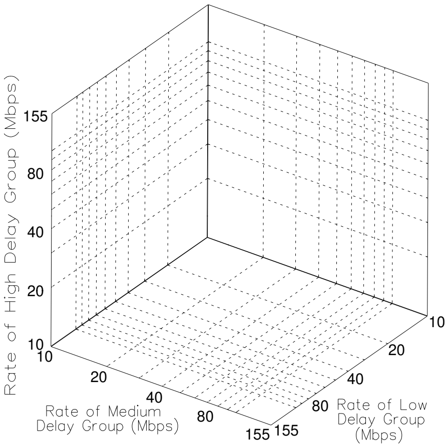


(b) EDF Scheduler.

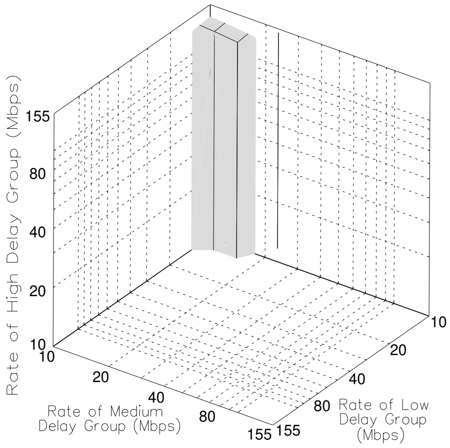


(c) SP Scheduler.

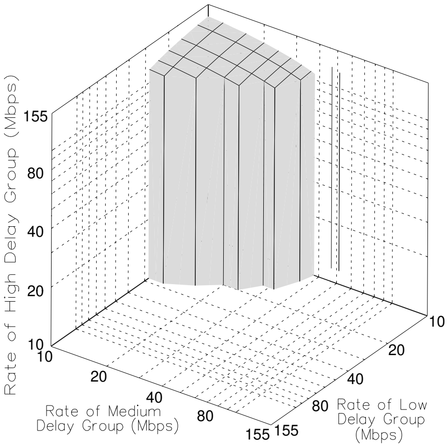
Fig. 9. Benchmark Schedulable Regions.



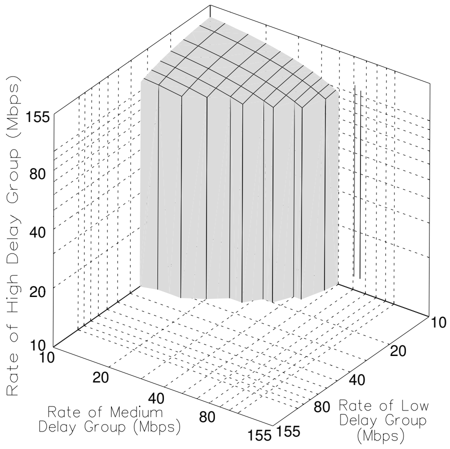
(a) RPQ at $\Delta = 12$ms.



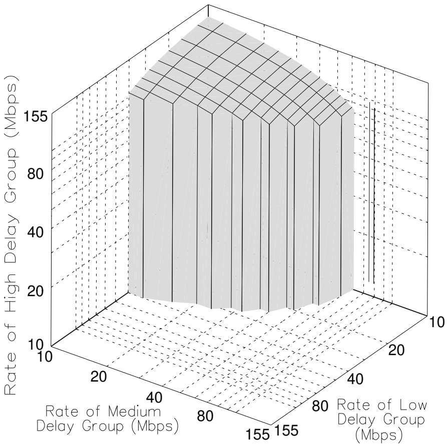
(b) RPQ at $\Delta = 6$ms.



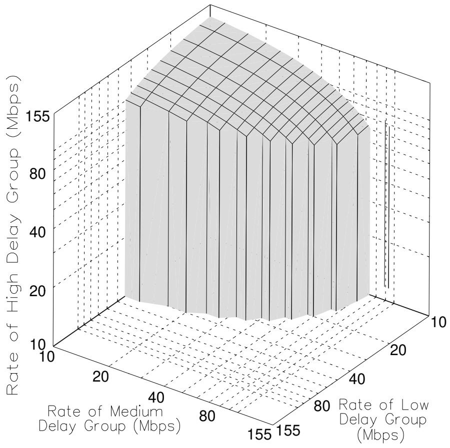
(c) RPQ at $\Delta = 4$ms.



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

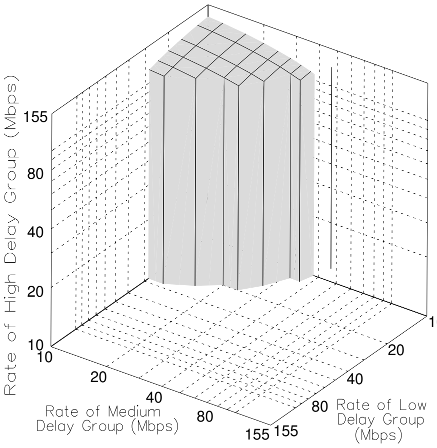


(e) RPQ at $\Delta = 2$ms.

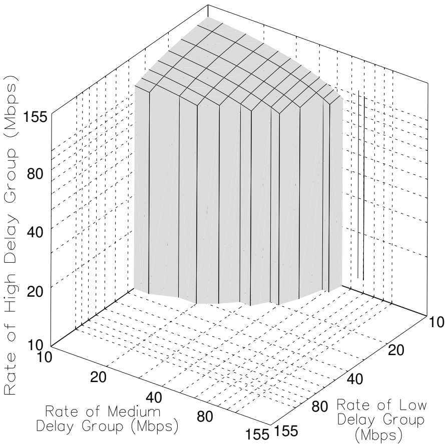


(f) RPQ at $\Delta = 1$ms.

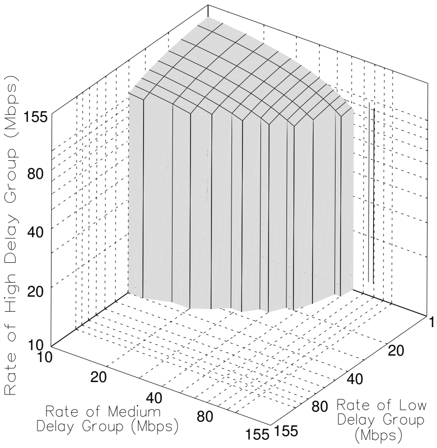
Fig. 10. Schedulable Regions for RPQ.



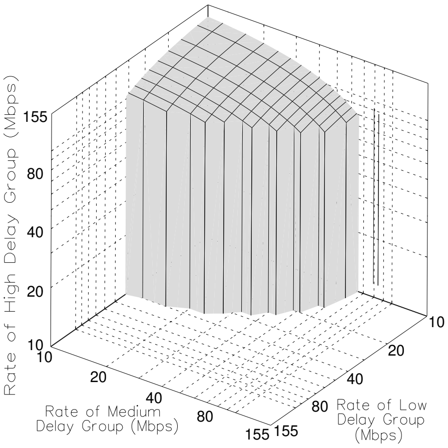
(a) RPQ$^+$ at $\Delta = 12$ms.



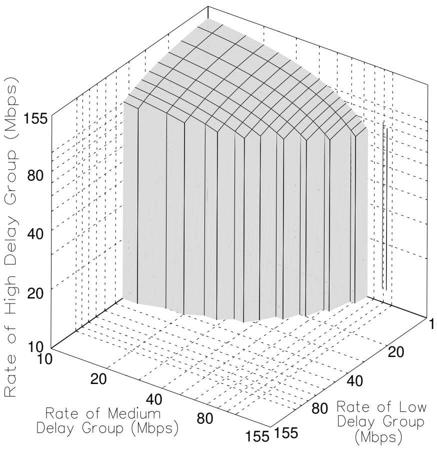
(b) RPQ$^+$ at $\Delta = 6$ms.



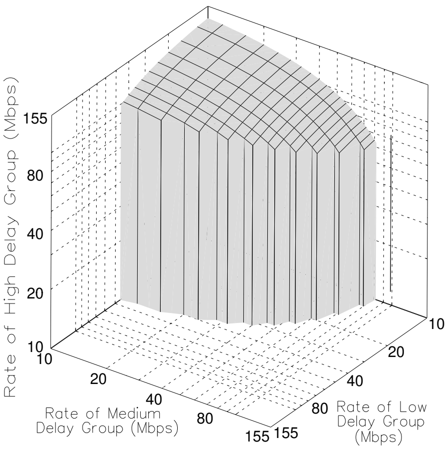
(c) RPQ$^+$ at $\Delta = 4$ms.



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



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



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

Fig. 11. Schedulable Regions for RPQ$^+$.

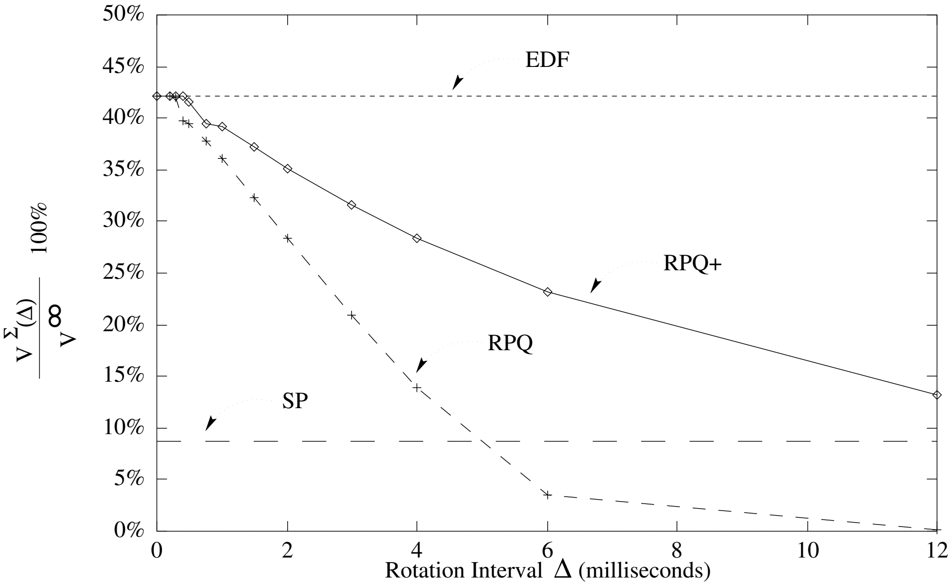
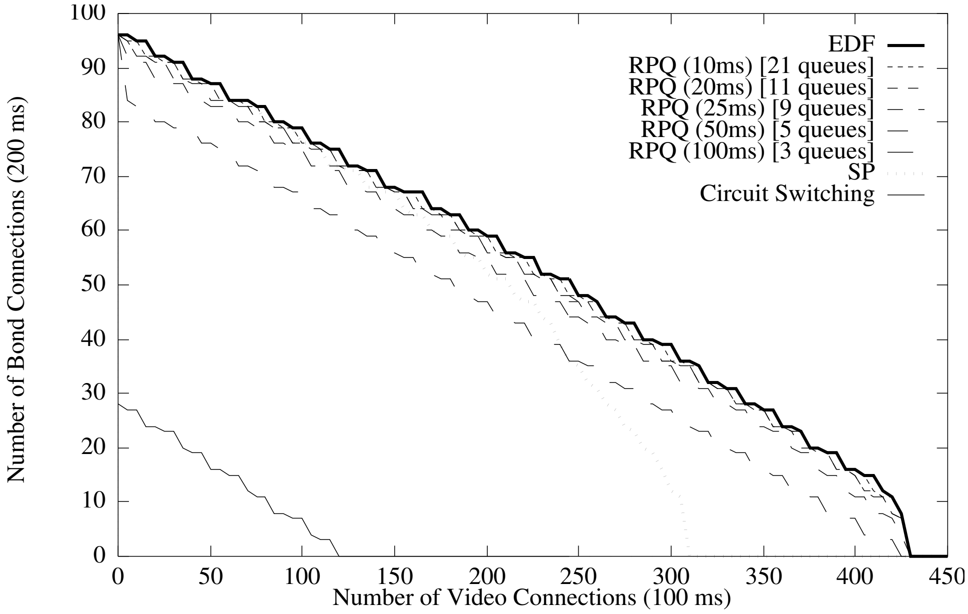
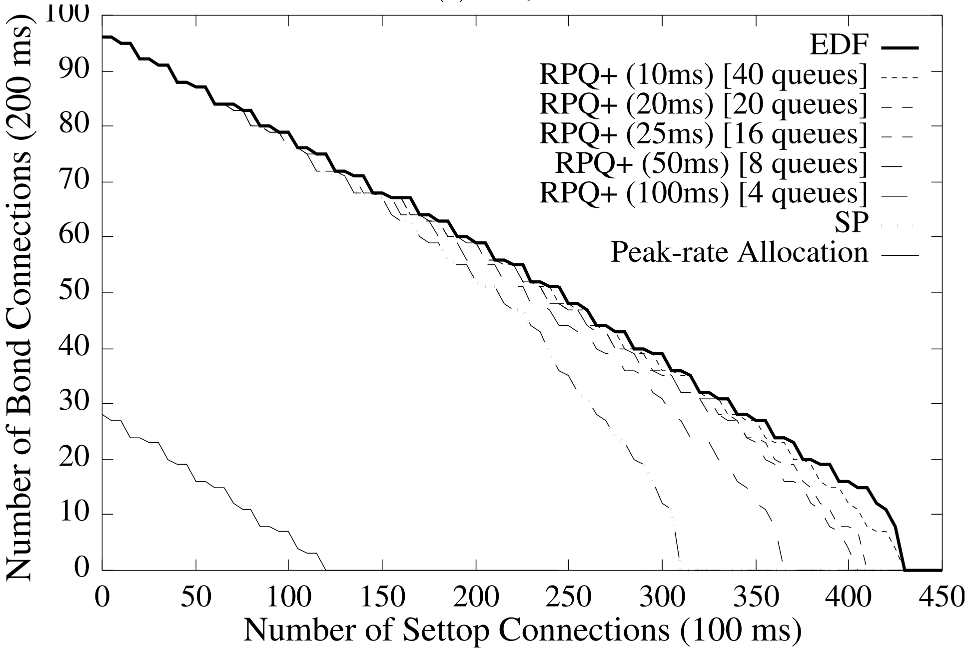


Fig. 12. Summary of Utilizations for Packet Schedulers.



(a) RPQ



(b) RPQ^{+}

Fig. 13. Example of RPQ and RPQ^{+} for MPEG Video Sequences.

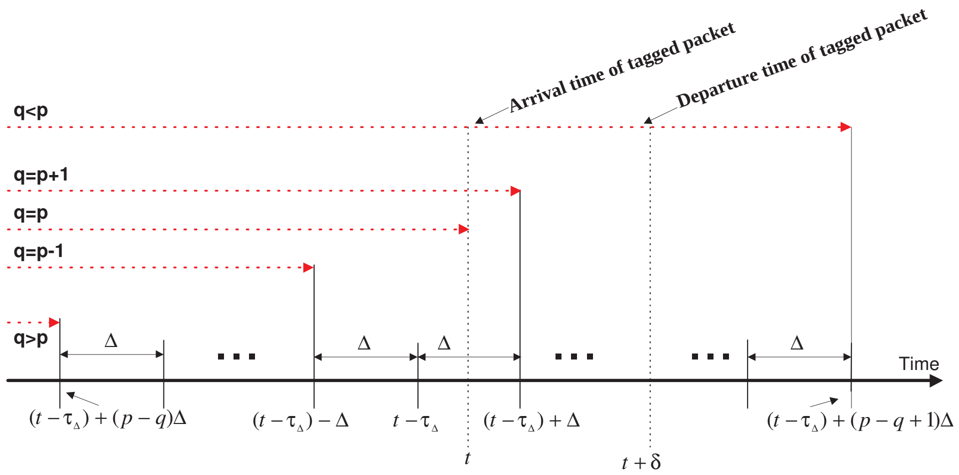


Fig. 14. Arrivals that are transmitted before the Tagged Packet.

Packet Scheduler	(Exact) Schedulability Conditions
EDF	For all $p = 1, 2, \ldots, P$: $$d_p \geq \frac{1}{R} \cdot \frac{\sum_{j \in \mathcal{C}_p} \sigma_p + \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} (\sigma_q - \rho_q d_q) + \max_{r>p} L_r^{max}}{1 - \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} \rho_q}$$
SP	For all $p = 1, 2, \ldots, P$: $$d_p \geq \frac{1}{R} \cdot \frac{\sum_{q=1}^{p} \sum_{q \in \mathcal{C}_q} \sigma_q + \max_{r>p} L_r^{max}}{1 - \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} \rho_q}$$
RPQ	For all $p = 1, 2, \ldots, P$: $$d_p \geq \frac{1}{R} \cdot \frac{\sum_{p \in \mathcal{C}_p} \sigma_p + \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} (\sigma_q - \rho_q d_q) + \chi_{p \neq 1}(1 - \rho_1)\Delta + \max_{q>p} L_q^{max}}{1 - \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} \rho_q}$$ with $\chi_{p \neq 1} = 1$ if $p \neq 1$, and 0 otherwise.
RPQ$^+$	For all $p = 1, 2, \ldots, P$: $$d_p \geq \frac{1}{R} \cdot \frac{\sum_{j \in \mathcal{C}_p} \sigma_p + \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} (\sigma_q - \rho_q d_q) + \max_{r \leq p}(\sum_{s=1}^{r-1} \Delta\rho_s + \max_{s>r} L_r^{max})}{1 - \sum_{q=1}^{p-1} \sum_{q \in \mathcal{C}_q} \rho_q}$$

TABLE I

SCHEDULABILITY CONDITIONS FOR EDF, SP, RPQ AND RPQ$^+$ FOR LEAKY-BUCKET CONSTRAINED SESSIONS.

	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 II

Parameter Set for Scheduler with 155 Mbps Transmission Rate.