Fair Scheduling With Tunable Latency: A Round-Robin Approach

Hemant M. Chaskar and Upamanyu Madhow, *Senior Member, IEEE*

***Abstract*—Weighted fair queueing (WFQ)-based packet scheduling schemes require processing at line speeds for tag computation and tag sorting. This requirement presents a bottleneck for their implementation at high transmission speeds. In this paper, we propose an alternative and lower complexity approach to packet scheduling, based on modifications of the classical round-robin scheduler. Contrary to conventional belief, we show that appropriate modifications of the weighted round-robin (WRR) service discipline can, in fact, provide tight fairness properties and efficient delay guarantees to multiple sessions. Two such modifications are described: 1) list-based round robin, in which the server visits different sessions according to a precomputed list which is designed to obtain the desirable scheduling properties and 2) multiclass round robin, a version of hierarchical round robin with controls designed for good scheduling properties. The schemes considered are compared with well-known WFQ schemes and with deficit round robin (a credit-based WRR), on the basis of desirable properties such as bandwidth guarantees, fairness in excess bandwidth sharing, worst-case fairness, and efficiency of latency (delay guarantee) tuning. The scheduling schemes proposed and analyzed here operate with fixed packet sizes, and hence can be used in applications such as cell scheduling in ATM networks, time-slot scheduling on wireless links as in GPRS air interface, etc. A credit-based extension of the proposed schemes to handle variable packet sizes is also possible.**

***Index Terms*—Quality of service, round robin, scheduling, weighted fair queueing.**

I. Introduction

THE scheduling scenario considered here arises when a number of packet streams, called sessions, share an output link at the switch. Each session maintains a separate queue of its packets waiting for access to the transmission link. Packet transmissions must be scheduled so as to achieve the various objectives such as guaranteed minimum bandwidth to each session, fair excess bandwidth sharing (proportional [1] or state-dependent fairness [2]), worst-case fairness [3], and efficient scaling of latency with the number of sessions [4]. Further, the schedulers should have low complexity of implementation.

Different weighted fair queueing (WFQ)-based schemes [such as packet generalized processor sharing (PGPS) [5], self-clocked fair queueing (SCFQ) [1], worst-case fair weighted fair queueing (WF2Q) [3], and the schemes based on the rate proportional server (RPS) framework [6], [7]] offer different subsets of these desirable features. However, the best among them (WF2Q) is difficult to implement [1], [3] at high transmission speeds due to its high complexity. The complexity of WF2Q arises from two main sources: updating the virtual clock and sorting of packet tags to schedule the new transmission. Lower complexity WFQ schemes (which still require tag sorting), such as the SCFQ and RPS-based schemes, may not possess all the above-mentioned scheduling properties. For example, SCFQ lacks worst-case fairness, and also has inefficient latency tuning characteristics [4] (see also Section II), i.e., in SCFQ, the latency of the session increases with the total number of sessions sharing the link, even if the fraction of total link bandwidth allocated to the session is kept unchanged. PGPS is not worst-case fair either [3], even though its complexity of implementation is as high as that of WF2Q. RPS-based schemes such as PGPS and frame fair queueing (FFQ), are not worst-case fair. Further, the short-term fairness of RPS-based schemes such as FFQ is much worse than that of SCFQ or WF2Q. A lower complexity WFQ-based scheme called WF2Q+, which possesses all the properties of WF2Q, has been recently proposed [8]. WF2Q+ relieves the complexity of updating virtual clock in WF2Q. However, the complexity of tag sorting still exists in WF2Q+. It is more pronounced in WF2Q+, since it has to do two independent sorts during each transmission slot, one on virtual finishing times to decide the next transmission, and the other on virtual starting times to update the virtual clock.

Much of the recent research in reducing the processing requirement of the scheduler [1], [6]–[8] has concentrated on modifying the basic WFQ paradigm [5]. An exception is the credit-based version of round robin, called deficit round robin (DRR), proposed in [9]. Though DRR has lower complexity than WFQ schemes, its short-term fairness properties are worse than that of WFQ schemes. DRR is also not worst-case fair, and it has inefficient latency tuning characteristics (see Section II).

In contrast to this, the approach in this paper is to devise modifications of the weighted round-robin (WRR) discipline that preserve the good scheduling properties of the best WFQ schemes, such as WF2Q and WF2Q+. Accordingly, two categories of schedulers, namely, list-based WRR (Section III-B) and multiclass WRR (Section III-C) are proposed in this paper. List-based

Manuscript received February 22, 2000; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor T. V. Lakshman. This paper was presented in part at the IEEE Globecom'99. This work was supported by the U.S. Army Research Office under Grants DAAG55-98-1-0219 and DAAD19-00-1-0567, and by the National Science Foundation under Grants EIA-0080134 and ANI-0220118 (ITR).

H. M. Chaskar was with the with the Department of Electrical and Computer Engineering, University of Illinois at Urbana Champaign, Urbana, IL 61801 USA. He is now with the Nokia Research Center, Burlington, MA 01803 USA (e-mail: hemant.chaskar@nokia.com).

U. Madhow is with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA (e-mail: madhow@ece.ucsb.edu).

Digital Object Identifier 10.1109/TNET.2003.815290

WRR is a generalization of classical WRR, while multiclass WRR is based on hierarchical round robin, along with some controls designed to obtain good scheduling properties. It is shown that, as regards various fairness properties and delay characteristics discussed above, the best among these WRR schemes achieve the performance of the best WFQ schemes, namely, WF2Q and WF2Q+. The proposed WRR-based schemes do not involve packet tags, and hence, have lower complexity of implementation than WFQ-based schemes. Note that the complexity of the proposed schemes is no more than that of DRR, even though they have scheduling properties comparable to WF2Q and WF2Q+.

In this paper, these WRR-based schemes are analyzed for fixed packet sizes. They are thus applicable for cell scheduling in ATM networks. They are also useful in certain other scheduling scenarios where the schedulable unit of bandwidth is intrinsically of a fixed size. Examples are time slot scheduling on wireless link in GPRS networks, and frame scheduling on the air interface in third generation CDMA wireless networks. A credit-based extension, as in DRR (which is a credit-based extension of classical WRR), of the proposed schemes can be used to handle variable-length packets, but it is not discussed in this paper.

The rest of the paper is organized as follows. Section II describes the desirable objectives of scheduling and the state of the art with regard to achieving these objectives. The new WRR-based schedulers are proposed and analyzed in Section III.

II. Preliminaries

Consider a total of N packet streams, called sessions, sharing an output link at the router. Each session maintains a separate queue of its packets waiting for access to the transmission link. Packet transmissions on the link must be scheduled so as to achieve the various objectives described in the following sections.

1) Guaranteed Minimum Bandwidth or Isolation: Every session i must have a (prenegotiated) guaranteed fraction ϕ_i of the link bandwidth, irrespective of the traffic offered by other sessions.

2) Fair Excess Bandwidth Sharing: Any excess bandwidth on the link is to be distributed among active sessions in a fair manner. Two commonly used excess bandwidth distribution laws are as follows.

a) Proportional Sharing: The excess bandwidth available is distributed among active (backlogged) sessions in proportion to their guaranteed bandwidth fractions ϕ_is [5]. Fairness in excess bandwidth distribution (called "proportional fairness") is measured by the service discrepancy [1] (denoted by $\epsilon_{i,j}[t, t+\tau]$) between any two sessions i and j over any interval of time $[t, t+\tau]$ during which both of them are *continuously backlogged* (i.e., have packets waiting for transmission or in transmission). Thus

$$\epsilon_{i,j}[t, t+\tau] = \left| \frac{S_i[t, t+\tau]}{\phi_i} - \frac{S_j[t, t+\tau]}{\phi_j} \right|$$

where $S_k[t, t+\tau]$ is the service offered (amount of traffic transmitted) to session k ($k = i, j$) during interval $[t, t+\tau]$. For satisfactory proportional fairness, $\lim_{\tau\to\infty}(\epsilon_{i,j}[t, t+\tau]/\tau) = 0$, so that as the comparison interval $[t, t+\tau]$ expands, the difference between the *average* normalized services offered to competing sessions vanishes. Further, $\epsilon_{i,j}[t, t+\tau]$ for a given scheduling scheme must be small in magnitude. For an idealized "fluid system" [5], $\epsilon_{i,j}[t, t+\tau]$ is identically zero. However, in practice, for any packetized system, $\epsilon_{i,j}[t, t+\tau]$ is bounded away from zero by $(1/2)((1/\phi_i) + (1/\phi_j))$ (see [1]). For all WFQ schemes and also for the versions of WRR proposed here, we have $\epsilon_{i,j}[t, t+\tau] \leq c_{i,j}$, where $c_{i,j}$ is a constant *independent* of (the length of) the comparison interval. In this paper, $c_{i,j}$ will be referred to as the "proportional fairness index" (denoted by η_{PF}) for that scheme. Note that two scheduling schemes having the same value for η_{PF} have identical fairness properties (short and long term).[1]

b) State-Dependent Sharing: Here the excess bandwidth available at any time t is distributed according to the state of the system at time t. For example, all the excess bandwidth available at time t could be allocated to the session with the largest normalized (by the guaranteed bandwidth fraction) backlog at that time [2]. Since all WFQ schemes are designed to achieve proportional fairness, a modification to state-dependent sharing, although not impossible [2], is not simple. On the other hand, WRR schemes, though designed for proportional fairness, naturally extend to incorporate state-dependent excess bandwidth sharing as well.

3) Worst-Case Fairness: This notion is introduced in [3]. WFQ schemes ensure that the service offered to any session i in actual system until any time t does not lag behind the service offered to it until that time in the corresponding hypothetical (and indirectly simulated) fluid system, by more than a constant. However, for some session i, it can lead the latter by a large amount followed by the interruption in the service to that session until the fluid service catches up with the actual service (see [3] for an example of such a phenomenon). This causes burstiness in the service offered to some sessions which is undesirable for the proper functioning of service rate measurement schemes employed at the router. The WF2Q scheme in [3] is designed to avoid such a burstiness in the offered service. The property that distinguishes WF2Q (and WF2Q+) from PGPS, RPS, and SCFQ is called worst-case fairness. Following [3], a scheduling discipline is called "worst-case fair" if for any session i, the delay encountered by a packet of that session arriving at any time t, is bounded above by $(Q_i(t)/\phi_i)+C_i$, where $Q_i(t)$ is the queue size of session i at time t, ϕ_i is the guaranteed throughput to session i (with the total link speed normalized to unity) and C_i is a constant *independent* of the number of other sessions sharing the transmission link. In this paper, C_i for a given scheduling scheme is called the "worst-case fairness index" (denoted by η_{WF}) for that scheme.

4) Efficient Latency Tuning Characteristics: One of the proposed frameworks for lossless transport of real time data with guaranteed delay is to regulate the session's traffic at the network edge by a leaky bucket regulator [10]–[12], and to guarantee a service curve, defined by latency and rate [6], at each of

[1]Sometimes it is convenient to assign weights w_is to sessions that are proportional to ϕ_is and normalize the offered service by weights rather than the guaranteed bandwidth fractions.

the intermediate nodes. It is then possible to guarantee an upper bound on the end-to-end delay. In analogy with the leaky bucket regulator which enforces an upper affine envelope on the allowable volume of traffic of the session, a latency-rate ($\mathcal{LR}$) server [6] guarantees a lower affine envelope on the service offered to the session at the network node. The service to session i at a network node is called the latency-rate service with the latency θ_i and the rate ϕ_i, if for any interval of time $[t,\ t+\tau]$ lying *entirely* in the busy period of session i, we have

$$S_i[t,\ t+\tau] \geq \max\{0,\ \phi_i(\tau - \theta_i)\}.$$

For all WFQ schemes and also for WRR schemes presented here, such a latency-rate service is tuned by allocating an appropriate fraction ϕ_i of the total bandwidth to session i. Based on the relation of latency to the assigned bandwidth fraction (called here "latency tuning characteristics"), these scheduling schemes fall into two categories.

a) Efficient Schedulers (PGPS, WF2Q, WF2Q+, RPS): In this, the latency θ_i of session i, which has a fraction ϕ_i of the link bandwidth assigned to it, is given by $\theta_i = (1/\phi_i) + 1$ [6, Lemma 6], where all packets are assumed to be of a fixed length, requiring one unit time for transmission.

b) Inefficient Schedulers (SCFQ, Classical WRR, DRR): To study the latency tuning characteristics for SCFQ, classical WRR (WRR0) and DRR, it is convenient to think in terms of session weights. Let w_i (a nonzero integer) denote the weight assigned to session i, and let $W = \sum_{i=1}^{N} w_i$. Thus, the fraction $\phi_i = w_i/W$ of the link bandwidth is assigned to session i. Then, for SCFQ, WRR0, and DRR, if there are $(W - w_i)$ sessions other than session i, each with weight 1, it is easy to construct traffic patterns for which a packet of session i that initiates the backlogged period for session i, has to wait for $(W - w_i)$ transmission slots before departing. This gives $\theta_i \geq W - w_i = W(1-\phi_i)$. Thus, θ_i also depends on the total number of sessions (via W) that share the link. If $W \to \infty$ (i.e., the total number of sessions increases) while keeping ϕ_i unchanged (i.e., the resource allocation of session i is unchanged), $\theta_i \to \infty$. Note that this is *not* the case with the efficient scheduling schemes (e.g., PGPS, WF2Q, WF2Q+, RPS). Secondly, for inefficient schedulers, the latency θ_i decreases only linearly with ϕ_i. In contrast, θ_i decreases inversely with ϕ_i for efficient schedulers. Due to these two factors, in order to set the desired latency θ_i, the fraction of bandwidth ϕ_i to be allocated to session i in SCFQ, WRR0 and DRR could be larger than that for PGPS, WF2Q, WF2Q+ or RPS. That is, SCFQ, WRR0 and DRR have inefficient latency tuning characteristics.[2]

5) Complexity of Implementation: The complexity of implementation of these scheduling schemes is an important consideration, since link transmission speeds have been increasing at a faster rate than memory and processing speed [13]. In all WFQ schemes, every new packet arrival is stamped with a tag and packets are transmitted in an increasing order of tags. The complexity of WFQ arises from the following sources, of which the second is present in *all* WFQ schemes.

a) Tracking the System State: For the computation of tags to be stamped on new arrivals, all WFQ schemes have to track (with time) the state of the system (called "virtual time" [1], [5] or "system potential" [6]). In PGPS and WF2Q, in the worst case, $O(N)$ events (where N is the total number of sessions) can occur during the transmission of any given packet, each of which causes the rate of increase of the virtual time to change. Each of these events invoke moderate amount of processing. This makes implementation of PGPS and WF2Q difficult in high-speed routers [1]. This drawback is removed in SCFQ and FFQ (which is an RPS-based scheme) by employing only an approximate state tracking. However, such a simplification results in some undesirable properties. In SCFQ, it results in inefficient latency tuning characteristics, and for FFQ, it causes some deterioration in short-term fairness. Of course, neither SCFQ nor FFQ are worst-case fair. The scheme that relieves the complexity of state tracking described above while maintaining the good scheduling properties of WF2Q is WF2Q+. However, as mentioned before, in WF2Q+, the complexity of tag sorting is more pronounced.

b) Tag Sorting: In *all* WFQ schemes, before scheduling any packet transmission, it is required to determine which session among N has the packet with the smallest tag. Such a tag sorting requirement may become a bottleneck at high transmission speeds. Due to this bottleneck, typical implementations may use an approximate tag sorting using "binning" [13].[3] Note that WF2Q+ requires tag sorting for updating the virtual clock as well.

In contrast, there is no requirement of packet tags in WRR schemes, thus relieving the complexity bottleneck at high transmission speeds.

A Note on Deficit Round Robin (DRR): A credit-based version of WRR scheme that can handle variable packet sizes is proposed in [9]. Its η_{PF} is larger than the lower bound of $(1/2)((1/w_i) + (1/w_j))$ and also than that for most WFQ schemes. In other words, its *short-term* fairness properties are worse than that of WFQ schemes. DRR is also *not* worst-case fair and it has inefficient latency tuning characteristics. A direct way to see this (although this can be shown for variable packet sizes also) is to note that DRR is equivalent to classical weighted round robin (WRR0) if the packet sizes are fixed and, hence, exhibits the scheduling properties of the latter. In contrast, our aim in this paper is to devise WRR schemes that offer the good scheduling properties of WFQ schemes.

III. WRR-Based Solution to Packet Scheduling

In the rest of this paper, all packets are assumed to have fixed length. Time-slotted transmission is assumed, with one transmission slot being equal to the transmission time of one packet. The unit of time is taken to be the duration of a transmission slot. New packet arrivals are assumed to occur at the beginning of time slots, and departures are assumed to occur just before the end of time slots. New arrivals at the beginning of a given

[2]This drawback of WRR0 is eliminated in the refined versions of WRR proposed here.

[3]The range of tag values is divided into b bins. Each new arrival is placed into an appropriate bin as per its tag value. For scheduling new packet transmission, *any* packet from the first bin is chosen.

time slot are included in the backlog for the corresponding sessions when making the scheduling decisions for that slot.

A. Classical WRR: WRR0

In WRR0, each session i has an integer weight w_i. The server visits all sessions in a predetermined order. When the server visits any session i, it serves the packets of session i, up to a maximum of w_i, in a first-come-first-served manner. Thus, the maximum length of the round-robin cycle is $W = \sum_{i=1}^{N} w_i$. WRR0 guarantees the fraction of link bandwidth $\phi_i = w_i/W$ to session i. Over any *partial* round-robin cycle, over which both sessions i and j are continuously backlogged, session i (j) can lead j (i) by at most w_i (w_j) packets. So

$$\left| \frac{S_i[t, t+\tau]}{w_i} - \frac{S_j[t, t+\tau]}{w_j} \right| \leq 1.$$

Thus, for WRR0, $\eta_{\mathrm{PF}} = 1$. It is much larger (hence worse) than the lower bound of $(1/2)((1/w_i) + (1/w_j))$ [1] and is also larger than the value $(1/w_i) + (1/w_j)$ attained by WFQ schemes such as PGPS, WF2Q, and SCFQ, especially when w_i and w_j are larger than 1. Further, WRR0 has inefficient latency tuning characteristics (see Section II) and it lacks worst-case fairness. For the latter, suppose a new packet of session i arrives at time t when the server has just crossed i, and suppose (for simplicity) the backlog of session i at time t [denoted by $Q_i(t)$] is a multiple of w_i. Then, this packet departs after a maximum of $W(Q_i(t)/w_i)+(W-w_i+1)$. Thus, $\eta_{\mathrm{WF}} \geq (W-w_i+1) \geq W(1-\phi_i)$, which depends on the total number of sessions (via W) and, hence, WRR0 is *not* worst-case fair.

Though WRR0 is devoid of many of the desirable properties exhibited by WFQ-type schedulers, its attractive feature is simplicity of implementation. In the following sections, we propose refined versions of the basic round-robin discipline to eliminate the undesirable properties of WRR0, while preserving its simplicity. (Note that DRR is a credit-based extension of WRR0.)

B. Generalization of WRR Discipline: List-Based WRR

In the generalization of classical WRR discipline, instead of serving w_i packets of session i in a single visit to session i, the service is distributed evenly over the round-robin cycle. For this a (periodic) list of session identities, called a "service list," is maintained. The number of times session i appears in the service list is proportional to its weight w_i, but these appearances are not necessarily consecutive as in WRR0. The server visits the sessions' queues according to this service list. It is important to note that the service list is updated only at the time of session termination or new session establishment, and *not* in every packet transmission slot. We now describe three such list-based WRR schemes and establish their scheduling properties.

1) Simply Interleaved WRR: WRR1: To compute the service list for WRR1, imagine that there are in all $W_{\max} = \max_i w_i$ bins. Session i with weight w_i registers itself in the first w_i bins. A service list is then computed by listing all the sessions in the first bin, followed by all those in the second bin, and so on, up to the $W_{\max}$th bin. The length (period) of the service list equals $W = \sum_{i=1}^{N} w_i$, which is also the maximum length of the round-robin cycle.

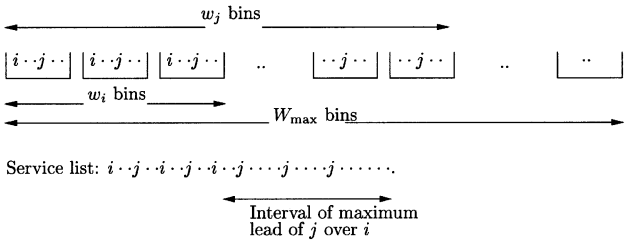


Fig. 1. Computing the service list for WRR1.

In order to calculate η_{PF} for WRR1, consider two sessions i and j with $w_j \geq w_i$. In any *partial* round-robin cycle, j can lead i by at most $w_j - w_i + 1$ packets. This maximum lead of j over i occurs when that partial round-robin cycle contains the $w_j - w_i + 1$ visits to j near the end of the service list and contains no visit to i (see Fig. 1). Hence

$$\frac{S_j[t, t+\tau]}{w_j} - \frac{S_i[t, t+\tau]}{w_i} \leq \frac{w_j - w_i + 1}{w_j}.$$

Also, due to the cyclic nature of service in any WRR discipline, the maximum *normalized* service by which j can lead i, is the same as the maximum *normalized* service by which i can lead j. In other words

$$\left| \frac{S_i[t, t+\tau]}{w_i} - \frac{S_j[t, t+\tau]}{w_j} \right| \leq \frac{w_j - w_i + 1}{w_j}.$$

Hence, for WRR1, $\eta_{\mathrm{PF}} = (w_j - w_i + 1)/w_j$, which is much smaller (short-term fairness is better) than that for WRR0, especially when w_i and w_j are both larger than 1 but approximately equal to each other.

2) Uniformly Interleaved WRR: WRR2: In WRR2, the total number of bins equals the least common multiple (denoted by W_{LCM}) of $\{w_i\}_{i=1}^{N}$. Session i registers itself in every $(n(W_{\mathrm{LCM}}/w_i))$th bin for $1 \leq n \leq w_i$. A service list is then computed, by listing the sessions bin after bin. The length (period) of the service list is $W = \sum_{i=1}^{N} w_i$, which is also the maximum length of round-robin cycle.

Near Optimality of Proportional Fairness of WRR2: For WRR2, $\eta_{\mathrm{PF}} = (1/w_i) + (1/w_j)$ (see the Appendix). This is no larger than that can be achieved by any of the WFQ schemes proposed so far. For PGPS [5], WF2Q [3], SCFQ [1], and WRR2, $\eta_{\mathrm{PF}} = (1/w_i) + (1/w_j)$, which is also near optimal in that it is within two times the corresponding value in *any* packetized system [1].[4] In other words, the proportional fairness (on any time scale, short or long) of WRR2 is as good as that of any WFQ scheme and is near optimal.

Drawbacks of WRR1 and WRR2: Though WRR1 and WRR2 have progressively better proportional fairness than that of WRR0 (with WRR2 having near optimal η_{PF}), both of them *lack* worst-case fairness. Also, neither of them have efficient latency tuning characteristics.[5] In WRR2, if session i has weight $w_i > 1$, and all of the other $W - w_i$ sessions each have weight 1, all of these other sessions register in the W_{LCM}th bin. Therefore, the $(w_i - 1)$th and the w_ith entries of session i in

[4]The proportional fairness index for FFQ [7] (an RPS-based WFQ scheme) is $(2/\min_i w_i) + \max(1/w_i, 1/w_j)$.

[5]Thus, as regards the scheduling properties, WRR2 is a weighted round-robin counterpart of SCFQ.

the service list will be separated by $W - w_i$ entries due to other sessions. In the above example, the same happens for WRR1, between the first and the second entries of session i in the service list. This causes latency $\theta_i = (W - w_i) = W(1 - \phi_i)$ and, hence, WRR1 and WRR2 exhibit inefficient latency tuning (see Section II). The same example also shows that WRR1 and WRR2 are not worst-case fair.

3) WF2Q Interleaved WRR—WRR3: This list-based WRR that has near-optimal proportional fairness, possesses worst-case fairness, and has efficient latency tuning characteristics was pointed out in [14]. In this, the service list is computed by assuming that all sessions are always backlogged and determining the sequence in which the packets are transmitted in WF2Q scheme [3]. The service list is then set equal to this sequence. Further details about this are omitted due to space limitations.

Finally, note that a paradigm somewhat similar to the list-based schedule was used in [15]. There, slot queues are used to keep track of the schedule, even if a packet is not in the queue. The packets are then scheduled from the packet queues according to the generated schedule.

Next, a scheme (referred to here as multiclass WRR) from the second category of WRR schedulers, namely, the category based on the hierarchical round robin, is described. It is shown that multiclass WRR offers all the scheduling properties of the best WFQ schemes, namely, WF2Q and WF2Q+.

C. Multiclass WRR

For initial exposition of multiclass WRR, consider the case of two classes of sessions, namely, $\mathcal{C}_1$ and $\mathcal{C}_2$. Let there be N_1 and N_2 sessions in classes $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. *All* sessions have weight 1. Multiclass WRR operates by embedding smaller round-robin "minicycles" within larger ones. Thus, let D_1 be the maximum length of the round-robin cycle in which all the sessions of class $\mathcal{C}_1$ must be visited, and let $D_2 > D_1$ denote the same for class $\mathcal{C}_2$. We assume that $D_2 = nD_1$ for an integer $n > 1$. Further, the following feasibility condition holds:

$$\frac{N_1}{D_1} + \frac{N_2}{D_2} \leq 1. \tag{1}$$

This is nothing but an alternate representation of the condition that $\sum_{i \in \mathcal{C}_1 \cup \mathcal{C}_2} \phi_i \leq 1$, since if session $i \in \mathcal{C}_1(\mathcal{C}_2)$, then it is served in the round-robin cycle of maximum duration D_1 (D_2), and, hence, the fraction ϕ_i of the link bandwidth assigned to it is $\phi_i = 1/D_1(1/D_2)$.

The maximum length of any minicycle is set to D_1 (the smallest among all D_is) visits. In every minicycle, all sessions in class $\mathcal{C}_1$ are always visited. After this, the sessions in class $\mathcal{C}_2$ are visited from the leftover visits until the length of the minicycle reaches the maximum of D_1 or the last session in class $\mathcal{C}_2$ is visited, whichever occurs first. The sessions in class $\mathcal{C}_2$ take turns at consuming the bandwidth left over in successive minicycles after sessions in class $\mathcal{C}_1$ have been visited. Rewriting (1), we have

$$N_2 \leq D_2 - \frac{D_2}{D_1} N_1 = n(D_1 - N_1). \tag{2}$$

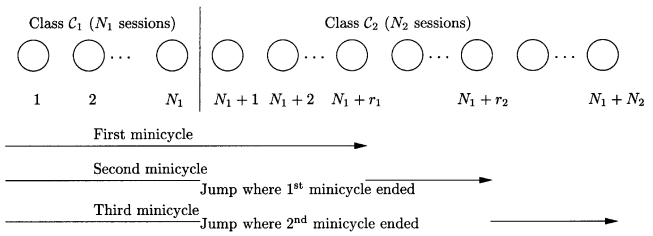


Fig. 2. Operation of multiclass WRR for two classes. $D_2 = 3D_1$ assumed.

Hence, at least N_2 visits are always available for the sessions in $\mathcal{C}_2$ over n successive minicycles.

The operation of multiclass WRR is shown in Fig. 2 for $n = 3$. In the figure, the first minicycle terminates after visiting the $(N_1 + r_1)$th session. During the second minicycle when the server crosses the class boundary, it *jumps* to visit the $(N_1 + r_1 + 1)$th session. The third minicycle ends (probably before D_1 sessions have been visited since its beginning) when the last session in class $\mathcal{C}_2$ is visited.

While the preceding is a classical hierarchical round-robin system, as described next, it is necessary to impose two controls on the dynamic evolution of the cycle in order to ensure good scheduling properties.

Control 1—Measuring the Length of Minicycle in Terms of Offered Service Opportunities Rather Than Transmissions: When the server visits some session i, but session i does not use this service opportunity because its queue is empty, such a *degenerate* visit to session i is still counted toward the length of the minicycle. The service is still *work conserving* because, if the offered service opportunity (henceforward referred to as a "visit") is not used by a particular session, the server moves on to the next backlogged session in order to transmit its packet during that time slot.

To illustrate the necessity of Control 1, in Fig. 3, suppose that sessions A, C, D, E, F are always backlogged, while session B is backlogged only the second transmission slot onwards. The sequence in which the packets are transmitted without and with Control 1 are shown in Fig. 3. The action of the control is indicated by the symbol "+." The visit to session B occurring at "+" is not used by session B, as its queue is empty at that time. So the server moves on to session C and transmits its packet during that transmission slot. The (degenerate) visit to session B occurring at "+" is still counted toward the length of the minicycle.

Note that when Control 1 is not used, the service to session E in the first $\mathcal{C}_2$ round-robin cycle "slips" ahead of its nominal (when all sessions are backlogged) position, while that in the second $\mathcal{C}_2$ round-robin cycle is in the nominal position. This causes the distance between the first and the second visits to session E to be 10, which is greater than $D_2 = 8$. The proposed control of measuring the length of the minicycle in terms of visits and *not* in terms of transmissions avoids such a slip, as it keeps the relative positions of visits to sessions unaffected by any session not using its service opportunity.

Control 2—Not Starting the New Round-Robin Cycle Too Early: According to this control, the sessions in class $\mathcal{C}_2$ are not allowed to obtain service opportunities (visits) for their mth round-robin cycle prior to the $[(m-1)(D_2/D_1) + 1]$th

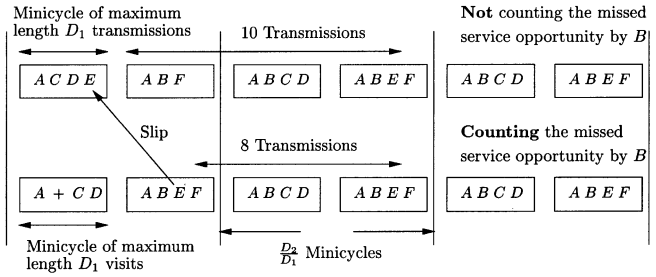


Fig. 3. Necessity of Control 1. $\mathcal{C}_1 = \{A, B\}$, $D_1 = 4$, and $\mathcal{C}_2 = \{C, D, E, F\}$, $D_2 = 8$.

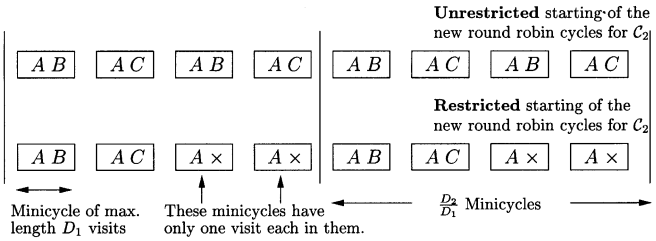


Fig. 4. Necessity of Control 2. $\mathcal{C}_1 = \{A\}$, $D_1 = 2$, and $\mathcal{C}_2 = \{B, C\}$, $D_2 = 8$.

minicycle. Thus, if they complete the $(m-1)$th round-robin cycle[6] in some minicycle prior to the $[(m-1)(D_2/D_1)]$th minicycle, then the leftover visits from class $\mathcal{C}_1$ are not offered to class $\mathcal{C}_2$ until in the $[(m-1)(D_2/D_1)+1]$th minicycle. The sessions in class $\mathcal{C}_2$ start obtaining such leftover visits, if any, for their mth round-robin cycle, starting from the $[(m-1)(D_2/D_1)+1]$th minicycle. This can be implemented by keeping a cyclic counter of length D_2/D_1 corresponding to class $\mathcal{C}_2$. The counter is incremented by one at the beginning of every new minicycle. After the termination of a given round-robin cycle of class $\mathcal{C}_2$, a new cycle is not allowed to start until the minicycle at the beginning of which the counter counts zero. Such a restriction is essential to make multiclass WRR proportionally fair, as illustrated in Fig. 4.

In Fig. 4, all sessions are assumed to be always backlogged and the sequence in which the packets are transmitted without and with Control 2 are shown. A leftover visit is made available by class $\mathcal{C}_1$ at the position indicated by "$\times$," but it is not offered to class $\mathcal{C}_2$ (and also not counted toward the length of minicycle), due to the action of Control 2. This causes early termination (before D_1 sessions have been visited) of the minicycle as there is no class after class $\mathcal{C}_2$ to use these visits. Note that when Control 2 is not in action, $r_A = (1/2)$, $r_B = 1/4$, and $r_C = 1/4$, where r_X denotes the average throughput of session X. Thus, $r_A/r_B = r_A/r_C = 2$, even if $\phi_A/\phi_B = \phi_A/\phi_C = (1/D_1)/(1/D_2) = 4$. On the other hand, with Control 2, $r_A/r_B = r_A/r_C = 4$ as desired.

Note that the positions indicated by symbols "$+$" and "$\times$" merely show the actions of the controls and *not* the transmission slots. The service is *work conserving* even after the inclusion of the preceding two controls.

The multiclass WRR service algorithm described above exhibits efficient latency tuning and worst-case fairness. These properties follow from the following lemma.

Lemma 3.1: In multiclass WRR for two classes $\mathcal{C}_1$ and $\mathcal{C}_2$, the distance between the successive visits to any session $i \in \mathcal{C}_k$ $(k = 1, 2)$ is no more than D_k.

Proof: The claim is trivially true if session $i \in \mathcal{C}_1$. So consider $i \in \mathcal{C}_2$. Let $\mathcal{J}_m$ (for $m \geq 1$) indicate the interval consisting of the minicycles $(m-1)(D_2/D_1)+1$ to $m(D_2/D_1)$. Then:

1) The length of any $\mathcal{J}_m$ is no more than D_2 visits.

2) From (2), at least N_2 visits are left over by class $\mathcal{C}_1$ during each of the $\mathcal{J}_m$s.

3) Relative positions of these visits left over by class $\mathcal{C}_1$ in any $\mathcal{J}_m$, with respect to the beginning of $\mathcal{J}_m$, are fixed (invariant with m), thanks to Control 1.

4) By virtue of Control 2, first N_2 of these visits left over by class $\mathcal{C}_1$ during any $\mathcal{J}_m$ are used to visit the sessions in class $\mathcal{C}_2$ starting from the first session in $\mathcal{C}_2$. □

Latency Tuning Characteristics (Two-Class Case): Due to Lemma 3.1, a packet of session $i \in \mathcal{C}_k$ that starts the backlogged period for i departs before a maximum of D_k transmission slots. Hence, the latency of session i is $\theta_i = D_k$. The fraction of total link bandwidth that is assigned to session i is $\phi_i = 1/D_k$. Hence, $\theta_i = 1/\phi_i$. Since latency scales inversely with the bandwidth fraction and is independent of the total number of sessions, multiclass WRR has efficient latency tuning characteristics (see Section II).

Worst-Case Fairness (Two-Class Case): Suppose a packet of session $i \in \mathcal{C}_k$ arrives at time t_a when the backlog of session i is $Q_i(t_a)$. Since the distance between the successive visits to session i is no more than D_k (Lemma 3.1), this packet departs *before* time $t_d = t_a + D_k[Q_i(t_a)+1] = t_a + (Q_i(t)/\phi_i) + (1/\phi_i)$. Hence, for multiclass WRR, $\eta_{\mathrm{WF}} = 1/\phi_i$ which is independent of the total number of sessions sharing the link and, hence, multiclass WRR is worst-case fair.[7]

Proportional Fairness (Two-Class Case): The proportional fairness index of two-class multiclass WRR is $\eta_{\mathrm{PF}} = (1/\phi_i) + (1/\phi_j)$, which is near optimal (see Section II). η_{PF} is derived directly for the general case (more than two classes) of multiclass WRR in Section III-C1.

To summarize the two-class case, we have shown that multiclass WRR has all the desirable scheduling properties. Its complexity is the same as that of WRR0. The only additional requirement is to keep track of the state in which the previous minicycle ended. This can be achieved by keeping a register at each class boundary pointing to the location of jump in that class. (When Control 2 prohibits the start of new round-robin cycle for some class, this jump location register contains an indication for jump *beyond* the class.)

1) Multiclass WRR for the General Case ($K \geq 2$ Classes): Consider $K \geq 2$ classes of sessions denoted by $\mathcal{C}_1$ to $\mathcal{C}_K$ with D_1 to D_K as the maximum lengths of their round-robin cycles respectively. It is assumed in the proofs of the scheduling properties of multiclass WRR that D_k divides D_{k+1} (denoted by $D_k \,|\, D_{k+1}$) for all $k = 1$ to $K-1$. The implementation of multiclass WRR, however, does *not* require any such condition. Moreover, our conjecture is that the scheduling properties proved here continue to hold *qualitatively*, even when the above-mentioned divisibility condition does

[6]Meaning that all sessions in class $\mathcal{C}_2$ have been *visited* $(m-1)$ times.

[7]η_{WF} of multiclass WRR is the same as that for the WF2Q scheme [3].

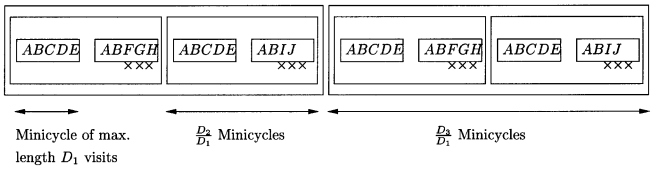


Fig. 5. Example of operation of multiclass WRR. $\mathcal{C}_1 = \{A, B\}$, $D_1 = 5$; $\mathcal{C}_2 = \{C, D, E\}$, $D_2 = 10$, and $\mathcal{C}_3 = \{F, G, H, I, J\}$, $D_3 = 20$.

not hold. Let N_k be the number of sessions in class $\mathcal{C}_k$. All sessions have weight 1. Then, on similar lines to the two-class case discussed earlier, the operation of multiclass WRR can be described as follows.

1) The feasibility condition must hold. Hence, $\sum_{k=1}^{K}(N_k/D_k) \leq 1$.

2) The maximum length of a minicycle $= \min_k D_k = D_1$ visits.

3) A new minicycle always starts from the first session in class $\mathcal{C}_1$. In any minicycle, the sessions in class $\mathcal{C}_k$ are visited from the leftover visits, if any, from classes $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{k-1}$.

4) Control 1 is operational as before, i.e., the length of a minicycle is measured in terms of visits (i.e., offered service opportunities) and not in terms of transmissions.

5) Control 2 is operational. According to this, the sessions in class $\mathcal{C}_k$ are not allowed to start their mth round-robin cycle prior to the $[(m-1)(D_k/D_1)+1]$th minicycle.[8]

The operation of multiclass WRR is shown by example in Fig. 5 for $K = 3$. The figure shows the sequence in which the sessions are *visited*. The symbol "$\times$" indicates the action of Control 2. For example, in the second minicycle in Fig. 5, three visits are left over by the class $\mathcal{C}_1$. However, they are not offered to class $\mathcal{C}_2$, as it is too early to start the new round-robin cycle of class $\mathcal{C}_2$. Hence, these three leftover visits are available to class $\mathcal{C}_3$ which uses them to visit sessions F, G and H.

It is shown below that multiclass WRR has efficient latency tuning characteristics, it is worst-case fair, and has near-optimal proportional fairness. The first steps toward showing these properties are the following two lemmas.

Lemma 3.2: Consider multiclass WRR for $K \geq 2$, as described above. Let

$$\sum_{k=1}^{K} \frac{N_k}{D_k} = 1 - \epsilon \qquad \text{for } \epsilon \geq 0.$$

Denote by $\mathcal{J}_{k,m}$ (for $k = 1$ to K and $m \geq 1$) an interval consisting of the minicycles $(m-1)(D_k/D_1)+1$ to $m(D_k/D_1)$. Fix any class $\mathcal{C}_k$ ($1 \leq k \leq K$). Then, during every $\mathcal{J}_{k,m}$, for $m \geq 1$:

1) All the sessions in class $\mathcal{C}_k$ are visited exactly once.

2) At least $\epsilon_k D_k$ visits are available for the use by classes $\mathcal{C}_{k+1}, \mathcal{C}_{k+2}, \ldots, \mathcal{C}_K$, where

$$\epsilon_k = \epsilon + \sum_{j=k+1}^{K} \frac{N_j}{D_j} = 1 - \sum_{j=1}^{k} \frac{N_j}{D_j}, \qquad \text{if } k < K \text{ and } \epsilon_K = \epsilon.$$

[8]When the divisibility condition does not hold, this is the $[\lceil (m-1)(D_k/D_1) \rceil + 1]$th minicycle.

(When $k = K$, since there is no class after $\mathcal{C}_K$, these visits are null causing early termination of the corresponding minicycles.)

Proof: The proof is based on induction on the the total number of classes in the system and the actions of Control 1 and Control 2. The claim is trivially true for $K = 1$ and from the two-class case discussed before, it is seen to be true for $K = 2$ also.

Suppose the claim is true for K (induction hypothesis). By virtue of Control 1, it is safe to assume that all the sessions are backlogged at all times, without changing the relative positions of visits to sessions. Further in any minicycle, the visits made available by class $\mathcal{C}_K$ to the downstream classes are necessarily wasted, causing early terminations of the corresponding minicycles, since there is no class downstream to class $\mathcal{C}_K$. During any $\mathcal{J}_{m,K}$, it is convenient to think of these wasted visits [denoted by $S_{K\to *}(\mathcal{J}_{m,K})$] as being used by a dummy class $*$. By induction hypothesis

$$S_{K\to *}(\mathcal{J}_{m,K}) \geq \epsilon_K W_K.$$

Now we show that the claim is true for multiclass WRR with $(K+1)$ classes. For this, consider

$$\sum_{k=1}^{K+1} \frac{N_k}{D_k} = 1 - \epsilon \leq 1.$$

This $(K+1)$-class system can be broken up into two parts—a K-class subsystem followed by class $\mathcal{C}_{K+1}$. Thus, in the $(K+1)$-class system, the $(K+1)$th class acts as a dummy class for the preceding K-class subsystem. Now consider an interval $\mathcal{J}_{m,K+1}$ for any $m \geq 1$.

In order to prove item 1 in the lemma, it is sufficient (after recalling the action of Control 2) to show that the total number of visits made available by the K-class subsystem to its dummy class $*$ [which is nothing but the $(K+1)$th class] over any $\mathcal{J}_{m,K+1}$ [denoted by $S_{K\to K+1}(\mathcal{J}_{m,K+1})$] is no less than N_{K+1}. For this, note that

$$\mathcal{J}_{m,K+1} = \bigcup_{i=1}^{W_{K+1}/W_K} \mathcal{J}_{i+(m-1)W_{K+1}/D_k, K}$$

and, hence

$$\begin{aligned} &S_{K\to K+1}(\mathcal{J}_{m,K+1}) \\ &\quad = \sum_{i=1}^{W_{K+1}/W_K} S_{K\to *}(\mathcal{J}_{i+(m-1)W_{K+1}/D_k, K}) \\ &\quad \geq \frac{W_{K+1}}{W_K}\,\epsilon_K W_K \text{ (by induction hypothesis)} \\ &\quad = \left[\epsilon + \frac{N_{K+1}}{W_{K+1}}\right] W_{K+1} \text{ (since in } (K+1)\text{-class system,} \\ &\qquad\qquad \epsilon_K = \epsilon + N_{K+1}/W_{K+1}) \\ &\quad = \epsilon W_{K+1} + N_{K+1} \\ &\quad \geq N_{K+1}. \end{aligned} \tag{3}$$

In order to prove item 2 in the lemma, we need to show that for the $(K+1)$-class system

$$S_{K+1\to *}(\mathcal{J}_{m,K+1}) \geq \epsilon_{K+1} W_{K+1}.$$

This follows from the fact that

$$\begin{aligned} &S_{K+1\to *}(\mathcal{J}_{m,\,K+1}) \\ &\quad = S_{K\to K+1}(\mathcal{J}_{m,\,K+1}) - N_{K+1} \text{ (due to Control 2)} \\ &\quad \geq \epsilon W_{K+1} \text{ [from (3)]} \\ &\quad = \epsilon_{K+1} W_{K+1} \text{ (since } \epsilon = \epsilon_{K+1}). \end{aligned}$$

□

Lemma 3.3: In multiclass WRR with $K \geq 2$ classes, for any session $i \in \mathcal{C}_k$, the distance between the successive visits to session i is no more than D_k.

Proof: The lemma can be proved by the following sequence of three arguments. Consider any $\mathcal{J}_{k,m}$ for $m \geq 1$.

1) The length of $\mathcal{J}_{k,m}$ is no more than $(D_k/D_1)D_1 = D_k$ number of visits to sessions.
2) From Lemma 3.2, all sessions in class $\mathcal{C}_k$ are visited exactly once during $\mathcal{J}_{k,m}$.
3) The position of visit to session i, relative to the beginning of $\mathcal{J}_{k,m}$, is invariant with m. This is because at the beginning of $\mathcal{J}_{k,m}$, all the classes upstream to and including class $\mathcal{C}_k$ are in the same state independent of m, by virtue of Lemma 3.2 and Control 2. This state is the following: at the beginning of the $[(m-1)(D_k/D_1)+1]$ th minicycle which is the first minicycle in $\mathcal{J}_{k,m}$, for every class upstream to and including $\mathcal{C}_k$, the next session to be visited in that class is the first session in that class. Also, the visit to such a session is allowed by Control 2 this minicycle onwards.

   Now since session i in class $\mathcal{C}_k$ is visited from the visits leftover in successive minicycles by the classes upstream to it and the sessions in class $\mathcal{C}_k$ that are ahead of it, the visit to session i in every $\mathcal{J}_{k,m}$ occurs at the same position relative to the beginning of $\mathcal{J}_{k,m}$ irrespective of the value of m. In other words, there is no "slip" as in Fig. 3. □

Latency Tuning Characteristics and Worst-Case Fairness of Multiclass WRR: Multiclass WRR, for the general case of $K \geq 2$, has efficient latency tuning characteristics and is worst-case fair. These properties follow from Lemma 3.3 on identical lines to the two-class case.

Proportional Fairness of Multiclass WRR: For multiclass WRR, it can be shown that

$$\epsilon_{i,j}[t,\, t+\tau] = \left| \frac{S_i[t,\, t+\tau]}{\phi_i} - \frac{S_j[t,\, t+\tau]}{\phi_j} \right| < \frac{1}{\phi_i} + \frac{1}{\phi_j}.$$

Hence, the proportional fairness index for multiclass WRR is $\eta_{\mathrm{PF}} = (1/\phi_i) + (1/\phi_j)$, which is a near-optimal value as discussed earlier.

To derive such an upper bound on the service discrepancy, note that if $i(j) \in \mathcal{C}_{k_i}(\mathcal{C}_{k_j})$, 1). The length of $\mathcal{J}_{k_l,m}$ $(l = i,\, j)$ in terms of *minicycles* is *exactly* D_{k_l}/D_1.

2) From Lemma 3.2, all sessions in class $\mathcal{C}_{k_l}$ are visited exactly once during $\mathcal{J}_{k_l,m}$.

3) It can be argued that (details omitted for brevity) the position of visit to session l relative to the beginning of $\mathcal{J}_{k_l,m}$ is invariant with m.

Due to these facts, the distance between the minicycles that contain successive visits to session l $(l = i,\, j)$ is exactly D_{k_l}/D_1 minicycles. Once this is observed, derivation of η_{PF} for multiclass WRR is identical to that for WRR2 (see proof of Lemma A.1 in the Appendix, with "bins" in the proof of Lemma A.1 replaced by "minicycles" for multiclass WRR.

An example is given below to provide intuition into how multiclass WRR achieves the preceding scheduling properties. In particular, the transmission schedule generated by multiclass WRR is compared with those generated by other well-known schemes. Consider a system in which there is one session A with weight 10, and five other sessions, B_1–B_5, each with weight 1. [In other words, $\phi_A = (1/2)$ and $\phi_{B_k} = 1/10$ for $k = 1 \ldots 5$.) Let us assume that all these sessions are backlogged at all times. The transmission schedules generated by WF2Q and multiclass WRR, which are worst-case fair, and PGPS, which is worst-case unfair, are shown below.

WF2Q: $A\ \underline{B_1}\ A\ \underline{B_2}\ A\ \underline{B_3}\ A\ \underline{B_4}\ A\ \underline{B_5}\ \ldots$
Multiclass WRR: $A\ \underline{B_1}\ A\ \underline{B_2}\ A\ \underline{B_3}\ A\ \underline{B_4}\ A\ \underline{B_5}\ \ldots$
PGPS: $A\ A\ A\ A\ A\ \underline{B_1\ B_2\ B_3\ B_4\ B_5}\ \ldots$

Here, the underlined slots indicate interruption in service to session A. Note that such an interruption is long in PGPS, compared to the evenly distributed service in WF2Q and multiclass WRR. In this example, for multiclass WRR, A belongs to a class whose maximum length of round-robin cycle is five times shorter than that for the class containing B_1, B_2, B_3, B_4, and B_5 (i.e., $\mathcal{C}_1 = \{A\}$, $D_1 = 2$, and $\mathcal{C}_2 = \{B_1, \ldots, B_5\}$, $D_2 = 10$).

To illustrate the latency tuning characteristics, it is necessary to consider the delay encountered by a session that becomes newly backlogged. Let sessions B_1 to B_5 be always backlogged, while A becomes backlogged from the fourth time slot. The transmission schedules generated by WF2Q, PGPS, and multiclass WRR, which have efficient latency tuning characteristics, and SCFQ, which has inefficient latency tuning characteristics, are shown below.

PGPS:
$B_1\, B_2\, B_3\, \underline{A}\, A\, B_4\, B_5\, A\, A\, A\, A\, A\, B_1\, B_2\, B_3\, B_4\, B_5\ \ldots$
WF2Q:
$B_1\, B_2\, B_3\, \underline{A}\, A\, B_4\, B_5\, A\, B_1\, A\, B_2\, A\, B_3\, A\, B_4\, A\, B_5\ \ldots$
Multiclass WRR:
$B_1\, B_2\, B_3\, \underline{A}\ B_4\, A\, B_5\, A\, B_1\, A\, B_2\, A\, B_3\, A\, B_4\, A\, B_5\ \ldots$
(Order of visits in multiclass WRR:
$A\, B_1\, A\, B_2\, A\, B_3\, A\, B_4\, A\, B_5\ \ldots$)
SCFQ:
$B_1\, B_2\, B_3\, \underline{B_4\, B_5\, A}\, A\, A\, A\, A\ \ldots$

Here, the length of the underline shows the latency experienced by session A. Note that in SCFQ, a packet of session A that arrives in the fourth time slot has to wait for the packets of B_4 and B_5, even though ϕ_A is much larger than ϕ_{B_k}s.

State-Dependent Excess Bandwidth Sharing: We discussed in detail the proportional fairness properties of the new WRR schemes. WRR schemes can be easily modified to incorporate "state-dependent excess bandwidth sharing." Such a modification is obtained as follows: if any round-robin cycle (for multiclass WRR, this is the one corresponding to $\max_k D_k$) terminates before its maximum allowable length in terms of number of transmissions, the new cycle is not started immediately. The current cycle is stretched to its maximum allowable length by assigning state-dependent transmission slots.

IV. Conclusion

We have proposed a number of packet schedulers based on modifications of the classical round-robin discipline that incur lower complexity than comparable WFQ-based schemes. Multiclass WRR, as well as the best list-based WRR scheme (WRR3), have all the good scheduling properties of the best WFQ schemes, namely, WF2Q and WF2Q+.

In this paper, the WRR schemes are described and analyzed for fixed packet sizes. Credit-based versions of these schemes could handle variable packet sizes. In such a credit-based scheme, credit would be distributed among different sessions in the same order as visiting different sessions in the proposed WRR schemes, and the session's packet is chosen for transmission as soon as the credit accumulated with the session exceeds the size of its head-of-line packet. Detailed analysis of such credit-based versions, as well as devising other versions of round robin that can operate with variable-length packets, are important topics of future research.

Appendix
Derivation of $\eta_{\rm PF}$ for WRR2

Lemma A.1: For WRR2, the proportional fairness index is given by $\eta_{\rm PF} = (1/w_i) + (1/w_j)$.

Proof: As described in Section II, $\eta_{\rm PF}$ is obtained by determining the bound on the normalized service discrepancy

$$\epsilon_{i,\,j}[t,\,t+\tau] = \left| \frac{S_i[t,\,t+\tau]}{w_i} - \frac{S_j[t,\,t+\tau]}{w_j} \right|$$

over any time interval $[t,\,t+\tau]$ during which both sessions i and j are continuously backlogged. For this, we calculate the maximum amount by which $S_i[t,\,t+\tau]/w_i$ can exceed $S_j[t,\,t+\tau]/w_j$ and *vice versa*. Due to the cyclic nature of service in any WRR discipline, these two maximum differences are equal and it suffices to determine any one of them, say, the maximum amount by which $S_i[t,\,t+\tau]/w_i$ can exceed $S_j[t,\,t+\tau]/w_j$.

Let $V_j(p,\,x)$ denote the number of visits to j between the pth and $(p+x)$th visits to i, for any positive integer p and nonnegative integer x. Let $V_j(x) = \inf_p V_j(p,\,x)$. If interval $[t,\,t+\tau]$ contains exactly $x+1$ visits to i, we have

$$\frac{S_i[t,\,t+\tau]}{w_i} = \frac{x+1}{w_i} \quad \text{and} \quad \frac{S_j[t,\,t+\tau]}{w_j} \geq \frac{V_j(x)}{w_j}. \tag{4}$$

We now find the lower bound on $V_j(x)$ which in turn will allow us to bound the service discrepancy $\epsilon_{i,\,j}[t,\,t+\tau]$. In WRR2, $x+1$ successive visits to i span $x(W_{\rm LCM}/w_i)+1$ bins. Let there be in all y visits to session j contained in these $x(W_{\rm LCM}/w_i)+1$ bins spanned by $(x+1)$ visits to session i. First, consider the case $y > 0$. Denote by α_1 the distance between the bins containing the first among these $(x+1)$ visits to i and the first among these y visits to j. Similarly, let α_2 be the distance between the bins containing the last of these $(x+1)$ and y visits to i and j respectively. Since the distance between the bins containing the successive visits to j is $W_{\rm LCM}/w_j$, we have

$$\alpha_1 + \left[(y-1)\frac{W_{\rm LCM}}{w_j} + 1 \right] + \alpha_2 = x\,\frac{W_{\rm LCM}}{w_i} + 1.$$

Rearranging this

$$y = x\,\frac{w_j}{w_i} - \frac{(\alpha_1+\alpha_2)}{(W_{\rm LCM}/w_j)} + 1. \tag{5}$$

To obtain a good lower bound, three cases are required to be considered.

Case 1: $\alpha_1 = 0$ *or* $\alpha_2 = 0$. This means that j is also in the bin containing the first *or* the last among the $x+1$ visits to i. Let $\alpha_1 = 0$ without loss of generality. In this case from (5)

$$y = x\,\frac{w_j}{w_i} - \frac{\alpha_2}{(W_{\rm LCM}/w_j)} + 1.$$

If j precedes i in every bin, the first among these y visits to j may not fall in the interval $[t,\,t+\tau]$, even though the first among the $(x+1)$ visits to i does. To account for this, we have to subtract 1 from y and obtain

$$V_j(p,\,x) \geq y - 1 = x\,\frac{w_j}{w_i} - \frac{\alpha_2}{(W_{\rm LCM}/w_j)} > x\,\frac{w_j}{w_i} - 1$$

where the last inequality follows from the fact that $\alpha_2 < (W_{\rm LCM}/w_j)$, by definition.

The other cases, namely, *Case 2* ($\alpha_1 = 0$ *and* $\alpha_2 = 0$) and *Case 3* ($\alpha_1 > 0$ *and* $\alpha_2 > 0$) and also the case of $y = 0$ can be handled in a similar fashion as *Case 1*. Hence, for $y \geq 0$ and all $p \geq 1$, we have $V_j(p,\,x) > x(w_j/w_i) - 1$ and so

$$V_j(x) = \inf_p V_j(p,\,x) > x\,\frac{w_j}{w_i} - 1. \tag{6}$$

It then follows from (4) and (6) that

$$\frac{S_i[t,\,t+\tau]}{w_i} - \frac{S_j[t,\,t+\tau]}{w_j} < \frac{1}{w_i} + \frac{1}{w_j}.$$

□

References

[1] J. Golestani, "A self-clocked fair queueing scheme for broadband applications," in *Proc. IEEE INFOCOM*, 1994, pp. 636–646.

[2] N. G. Duffield, T. V. Lakshman, and D. Stiliadis, "On adaptive bandwidth sharing with rate guarantees," in *Proc. IEEE INFOCOM*, 1998, pp. 1122–1130.

[3] J. C. R. Bennett and H. Zhang, "WF2Q : Worst-case fair weighted fair queueing," in *Proc. IEEE INFOCOM*, 1996, pp. 120–128.

[4] D. Stiliadis and A. Varma, "Latency-rate servers: A general model for analysis of traffic scheduling algorithms," *IEEE/ACM Trans. Networking*, vol. 6, pp. 611–624, Oct. 1998.

[5] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Trans. Networking*, vol. 1, pp. 344–357, June 1993.

[6] D. Stiliadis and A. Verma, "Rate-proportional servers: A design methodology for fair queueing algorithms," *IEEE/ACM Trans. Networking*, vol. 6, pp. 164–174, Apr. 1998.

[7] ——, "Efficient fair queueing algorithms for packet-switched networks," *IEEE/ACM Trans. Networking*, vol. 6, pp. 175–185, Apr. 1998.

[8] J. C. R. Bennet and H. Zhang, "Hierarchical packet fair queueing algorithms," *IEEE/ACM Trans. Networking*, vol. 5, pp. 675–689, Oct. 1997.

[9] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round-robin," *IEEE/ACM Trans. Networking*, vol. 4, pp. 375–385, June 1996.

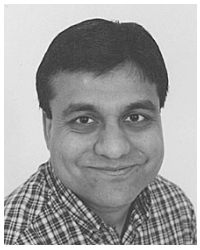
[10] R. Cruz, "A calculus of network delay, Part I: Network elements in isolation," *IEEE Trans. Inform. Theory*, vol. 37, pp. 114–131, Jan. 1991.

[11] ——, "A calculus of network delay—Part II: Network analysis," *IEEE Trans. Inform. Theory*, vol. 37, no. 1, pp. 132–141, January 1991.

[12] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case," *IEEE/ACM Trans. Networking*, vol. 2, pp. 137–150, Apr. 1994.

[13] V. P. Kumar, T. V. Lakshman, and D. Stiliadis, "Beyond best effort: Router architectures for the differentiated services of tomorrow's Internet," *IEEE Commun. Mag.*, vol. 36, pp. 152–164, May 1998.

[14] V. Bharghavan and S. Lu, private communication, Dec. 1998.
[15] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *IEEE/ACM Trans. Networking*, vol. 7, pp. 473–489, Aug. 1999.



Hemant M. Chaskar received the M.Eng. degree in electrical communication engineering from the Indian Insitute of Science, Bangalore, India, in 1995 and the Ph.D. degree in electrical engineering from the University of Illinois, Urbana-Champaign, in 1999.

He has been with the Nokia Research Center, Burlington, MA, since 1999, where he is involved in R&D activities in protocols, network architectures, and services for wireless networks. He has published numerous research articles in technical journals and conferences, and has served as expert panelist, tutorial lecturer, and program committee member in technical conferences. He actively paticipates in the Internet Engineering Task Force (IETF) activities. He has a number of granted patents in the area of wireless networking.

Dr. Chaskar is on the Open Mobile Services subcommittee of the IEEE Technical Committee on Personal Communication (TCPC).

Upamanyu Madhow (S'86–M'90–SM'96) received the Bachelor's degree in electrical engineering from the Indian Institute of Technology, Kanpur, in 1985 and the M. S. and Ph. D. degrees in electrical engineering from the University of Illinois, Urbana-Champaign, in 1987 and 1990, respectively.

From 1990 to 1991, he was a Visiting Assistant Professor at the University of Illinois. From 1991 to 1994, he was a Research Scientist with Bell Communications Research, Morristown, NJ. From 1994 to 1999, he was with the Department of Electrical and Computer Engineering, University of Illinois, first as an Assistant Professor, and since 1998, as an Associate Professor. Since December 1999, he has been an Associate Professor with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, where he is currently a Professor. His research interests are in communication systems and networking, with current emphasis on wireless communication.

Dr. Madhow is a recipient of the National Science Foundation CAREER Award. He has served as an Associate Editor for Spread Spectrum for the IEEE TRANSACTIONS ON COMMUNICATIONS, and as an Associate Editor for Detection and Estimation for the IEEE TRANSACTIONS ON INFORMATION THEORY.