



Available at
www.ElsevierComputerScience.com

POWERED BY SCIENCE @ DIRECT®

Computer Communications 27 (2004) 664–678

computer
communications

www.elsevier.com/locate/comcom

On the latency and fairness characteristics of pre-order deficit round Robin[☆]

Salil S. Kanhere, Harish Sethu*

Department of ECE, Drexel University 3141 Chestnut Street, Philadelphia, PA 19104, USA

Received 18 February 2003; revised 26 November 2003; accepted 2 December 2003

Abstract

In the emerging high-speed packet-switched networks, fair packet scheduling algorithms in switches and routers will form an important component of the mechanisms that seek to satisfy the Quality of Service (QoS) requirements of various applications. The latency bound of a scheduling discipline is an important QoS parameter, especially for real-time playback applications. Frame-based schedulers such as Deficit Round Robin (DRR), though extremely efficient with an $O(1)$ dequeuing complexity, lead to high latencies due to bursty transmissions of each flow's traffic. In recent work by Tsao and Lin [Computer Networks 35 (2001) 287], the authors propose a novel scheme, called Pre-order DRR, which overcomes this limitation of DRR while still preserving a low work complexity. In Pre-order DRR, a priority queue module, appended to the original DRR scheduler, re-orders the packet transmission sequence and thus distributes the output more evenly among flows, reducing burstiness and improving the latency. In this paper, we use a novel technique to analytically derive the latency bound of Pre-order DRR and we prove that our bound is a tight one. Our latency bound is significantly lower than the bound derived by Tsao and Lin, demonstrating that Pre-order DRR has even better performance characteristics than previously argued by its own authors. We further study the fairness properties of Pre-order DRR using a recently proposed measure of instantaneous fairness that seeks to more accurately capture the fairness of a scheduler. Using real router traffic traces, we present simulation results that demonstrate that Pre-order DRR achieves better fairness characteristics than other scheduling disciplines of equivalent complexity.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Packet scheduling; Latency; Fairness; Quality of service; Deficit round Robin

1. Introduction

Future high-speed packet-switched networks are expected to support a variety of services beyond the best-effort service available in the Internet today. A number of new applications such as distance learning and multimedia tele-conferencing rely on the ability of the network to guarantee such services. For example, such applications would expect the network to ensure that each flow of traffic receives its fair share of the bandwidth and is able to provide performance guarantees such as an upper bound on the end-to-end delay. This requires a Quality of Service (QoS)

mechanism to efficiently apportion, allocate and manage limited resources amongst competing users. An important component of such a mechanism is the traffic scheduling algorithm typically used at the output links of switches and routers.

The function of a packet scheduler at an output link is to select the next packet for transmission from among the packets awaiting transmission through the output link. Some of the most important and desirable properties of a scheduling discipline are fairness, efficiency, and low latency. Schedulers such as Weighted Fair Queueing [2,3], Worst-Case Fair Weighted Fair Queueing (WF^2Q) [4] and Self-Clocked Fair Queueing (SCFQ) [5] achieve good fairness through maintaining a global variable known as the virtual time or the system potential function. Such schedulers, known as sorted-priority schedulers, then use this variable to compute the timestamp for each packet indicating the relative priority of the packet for transmission over the output link. While they achieve good fairness and

[☆] This work was supported in part by NSF CAREER Award CCR-9984161. A preliminary version of this paper appeared in *Proceedings of the IEEE Conference on Local Computer Networks*, Tampa, Florida, USA, November 6–8, 2002.

* Corresponding author. Tel.: +1-215-8955-876; fax: +1-215-8951-695.
E-mail addresses: sethu@ece.drexel.edu (H. Sethu), salil@ece.drexel.edu (S.S. Kanhere).

low latencies, they are not very efficient due to the complexity of computing the system virtual time and the complexity of maintaining a sorted list of packets based on their timestamps.

In frame-based schedulers such as Deficit Round Robin (DRR) [6], on the other hand, no global virtual time is maintained and the scheduler simply visits all the non-empty queues in a round robin order. This reduces the per-packet work complexity of DRR to $O(1)$ with respect to the number of flows, making it attractive for implementation in routers, and especially so, in hardware switches. However, such schedulers do not achieve fairness comparable to that of timestamp-based schedulers. Round-robin schedulers such as DRR achieve good efficiency, but suffer from high startup latencies, burstiness in its output and delayed correction of unfairness. These weaknesses stem from the round robin nature of the service order and from the fact that each flow receives its entire share of service in the round at once in one service opportunity.

In Ref. [1], Tsao and Lin have proposed a new scheduling discipline called Pre-order Deficit Round Robin (Pre-order DRR) which aims at overcoming the aforementioned drawbacks. In Pre-order DRR a limited number of priority queues, p , are added to the DRR scheduler. These queues reorder the transmission sequence of the packets in each DRR round and thus eliminate the strict round-robin nature of service order. It is shown in Ref. [1] that Pre-order DRR belongs to the general class of Latency-Rate (\mathcal{LR}) servers [7] and the authors derive an upper bound on its latency. In this paper, we use a different, unique, and novel approach to analytically re-derive the latency bound of Pre-order DRR and we prove that our bound is a tight one. Our approach is based on interpreting the Pre-order DRR bandwidth allocations as an instance of the Nested Deficit Round Robin (Nested DRR) discipline discussed in Ref. [8]. The latency bound of Pre-order DRR derived in this paper is significantly lower than the bound derived by Tsao and Lin, demonstrating that Pre-order DRR has even better performance characteristics than previously argued by its own authors. The combination of high efficiency and a low latency bound makes Pre-order DRR an attractive scheduling discipline for guaranteed-rate services.

The rest of the paper is organized as follows. Section 2 presents a brief overview of DRR and Pre-order DRR. Section 3 discusses the interpretation of Pre-order DRR bandwidth allocations as an instance of allocations in Nested DRR. In Section 4, we present our analysis of the latency bound of Pre-order DRR. This section also includes a detailed comparison of the latency bound derived in this paper and that derived by Tsao and Lin in Ref. [1]. In Section 5, we compare the fairness properties of Pre-order DRR against other scheduling disciplines based on the Gini index, a new measure of fairness proposed in Ref. [9]. We present simulation results using real multimedia traffic traces to show that Pre-order DRR has superior fairness

properties than other efficient scheduling algorithms. Finally, Section 6 concludes the paper with a summary comparison of $O(1)$ packet schedulers. We also present a brief discussion on the implementation aspects of the Pre-order DRR scheduler and suggest a few improvements.

2. DRR and Pre-order DRR

Let r be the transmission rate of the output link, the access to which is controlled by a DRR scheduler. Assume that there are a total of n flows multiplexed on this link. Let ρ_i be the reserved rate for flow i and let ρ_{\min} be the minimum reserved rate among all the n flows. Since all these n flows share the same output link, $\sum_{i=1}^n \rho_i \leq r$. In order that the flows receive service proportional to their reserved rates, each flow i is assigned a weight, w_i , given by,

$$w_i = \frac{\rho_i}{\rho_{\min}} \quad (1)$$

Note that for any flow i , $w_i \geq 1$.

A flow is said to be *active* during a certain time interval, if it always has packets awaiting service in this interval. The DRR scheduler operates in terms of rounds wherein a *round* refers to one round robin iteration during which the DRR scheduler visits all the flows that are active at the instant that the round begins. The DRR scheduler allocates a *quantum*, Q_i to each active flow i , which is defined as the ideal service that flow i should receive in each round. The quanta assigned to the flows are in proportion to their reserved rates. Let Q_{\min} represent the quantum assigned to the flow with the lowest reserved rate. Hence, Q_i is given by $w_i Q_{\min}$. Let M denote the size of the largest packet that may potentially arrive during the execution of a scheduling algorithm. For the DRR scheduler to have a work complexity of $O(1)$, it is necessary that Q_{\min} is greater than or equal to M . A DRR frame is defined as the sum of the quanta allocated to all the active flows in a DRR round. Note that during a certain service opportunity, a flow may not be able to transmit a packet because doing so would cause the flow to exceed its allocated quantum. In such a case, the scheduler records the remainder of the quantum in the *deficit count* associated with the flow. This deficit is added to the quantum in the subsequent round. Hence a flow that does not receive its fair share of the bandwidth during a certain round is given an opportunity to receive proportionately more service in the next round. Let m represent the size of the largest packet that actually arrives during the execution of a scheduling algorithm. Note that $m \leq M$. It has been proved in Ref. [6] that,

$$0 \leq DC_i(s) \leq m - 1 \quad (2)$$

However, in all frame-based schedulers including DRR, each flow is served for a continuous period of time in proportion to its weight resulting in a highly bursty packet stream at the output of the scheduler. Also, due to the round

robin order of service, a flow that is lagging in service in comparison to other flows has to wait for its turn in the next round to compensate for the service lag. Further, there is no means for a lagging flow to receive precedence over all the other flows.

In Ref. [1], Tsao and Lin have proposed a new scheduling discipline called Pre-order DRR which aims to eliminate the above weaknesses of the DRR scheduler while trying to preserve its good properties such as its low work complexity. The assignment of the weights and the quanta in Pre-order DRR are identical to those in DRR. In fact, the Pre-order DRR scheduler also works in rounds. However, unlike the DRR scheduler which serves the active flows in a round robin fashion, the Pre-order DRR scheduler reorders the transmission sequence of the packets within each DRR round. In this section, we present a brief overview of the Pre-order DRR scheduling discipline. A significantly more detailed treatment may be found in Ref. [1].

Let us assume that a total of y packets are transmitted from flow i in the s -th round of service. The packets are labeled as $1, 2, \dots, y$ indicating their position in the stream of packets that are scheduled from flow i in round s . Note that y represents the last packet that is served in round s from flow i . As in DRR, the deficit count serves as a measure of past unfairness. Let $DC_i^m(s)$ represent the deficit count of flow i following the transmission of the m -th packet of the s -th round.

Definition 1. Define the *Quantum Availability*, denoted by $QA_i^m(s)$, of flow i after the transmission of the m -th packet from flow i in round s as follows:

$$QA_i^m(s) = \frac{DC_i^m(s)}{Q_i} \quad (3)$$

The *Quantum Availability* of a flow keeps track of the unused quantum of the flow in the current round.

In Pre-order DRR a priority queue module consisting of p queues and a *classifier* module are appended to the original DRR architecture. Let PQ_1, PQ_2, \dots, PQ_p represent the priority queues in the descending order of priority with PQ_1 as the highest priority queue and PQ_p denoting the lowest priority queue. Just as in DRR, the Pre-order DRR maintains a linked list of active flows called the *ActiveList*. However, the flows in the *ActiveList* are not served in a round robin manner as in DRR. This is a list of the active flows that have already received their fair share of service in the current round. These flows are, however, eligible for receiving service in the subsequent round. At the start of a round, the *Classifier* module classifies the packets that will be served in the current round from each flow present in the *ActiveList* according to its *Quantum Availability* into the p priority queues. In general, the priority queue, $z_i^m(s)$ into which the m -th packet served from flow i in the s -th round is added is calculated as follows,

$$z_i^m(s) = p - \lfloor QA_i^m(s) \times p \rfloor \quad (4)$$

Once all the packets that can be scheduled in the current round from flow i have been transferred from the flow buffers into the priority queues, if flow i is still active, it is added to the tail of the *ActiveList*.

When the scheduler is ready to transmit, it begins serving the packet at the head of the highest non-empty priority queue. Note that, if a packet is added to a priority queue that has a higher priority than the queue from which the scheduler is currently serving a packet, then following the current transmission, the scheduler will first serve the packet added into the higher priority queue. The round in progress ends when all the priority queues are empty. It has been proved in Ref. [1] that Pre-order DRR has a low worst-case work complexity of $O(1)$ with respect to the number of flows and $O(\log p)$ with respect to the number of priority queues.

3. The Nested DRR interpretation

The primary goal of the Pre-order DRR scheduler is to break the quantum allocated to a flow in a DRR round into several pieces so that it can be utilized in pieces over the course of the round. The Nested DRR scheduler proposed in Ref. [8] tries to eliminate the drawbacks of the DRR scheduler by creating a set of multiple rounds inside each DRR round and executes a modified version of the DRR algorithm within each of these inner rounds. The Nested DRR scheduler tries to serve Q_{\min} worth of data from each flow during each inner round. During an outer round, a flow is considered to be eligible for service in as many inner rounds as are required by the scheduler to exhaust its quantum. This results in a significantly lower latency bound, while preserving the $O(1)$ work complexity and the fairness characteristics of DRR. We can hypothetically interpret the operation of the Pre-order DRR scheduler as a *nested* version of DRR similar to Nested DRR. This interpretation is useful in analyzing the latency bound of the Pre-order DRR scheduler. Each round in DRR can be referred to as an *outer round*. The time period during which the Pre-order DRR scheduler serves the flows present in the priority queue PQ_u during the s -th outer round is referred to as *inner round* (s, u) . Thus, each outer round can be split into as many inner rounds as the number of priority queues, p . Since the Pre-order DRR scheduler visits the priority queues in a descending order of priority starting at priority queue PQ_1 and ending with queue PQ_p , the first and last inner rounds in outer round s are $(s, 1)$ and (s, p) , respectively.

The quantum assigned to each flow is divided equally among the p priority queues. Thus, the quantum allocated to flow i in each of its inner rounds is equal to Q_i/p . Let $Served_i(s, u)$ represent the total data scheduled from flow i in inner round (s, u) . Also let $DC_i(s, u)$ denote the deficit round of flow i at the end of the (s, u) -th inner round. Note that the deficit count of a flow at the end of the last inner round of

an outer round is the same as its deficit count at the end of the corresponding round in DRR. Also, this deficit count is carried over to the first inner round of the subsequent outer round. Hence,

$$DC_i(s, p) = DC_i(s) = DC_i(s + 1, 0)$$

Note that $DC_i(s + 1, 0)$ is used to represent the deficit count of flow i at the start of the inner round $(s + 1, 1)$. As in DRR, the deficit count is calculated as follows,

$$DC_i(s, u) = \frac{Q_i}{p} + DC_i(s, u - 1) - Served_i(s, u) \quad (5)$$

It can be easily proved that Eq. (2) which represents the bounds on the deficit count, $DC_i(s)$, also holds true for $DC_i(s, u)$. Hence for any flow i and inner round (s, u) ,

$$0 \leq DC_i(s, u) \leq m - 1 \quad (6)$$

In DRR, since the quantum of each flow is greater than or equal to the size of the largest packet that may potentially arrive during its execution, the scheduler is guaranteed to serve at least one packet from each of the active flows in each round. However, in Pre-order DRR, it may be possible that the sum of Q_i/p and $DC_i(s, u - 1)$ is less than the size of the packet at the head of flow i . In this case, flow i will not receive any service in inner round (s, u) . Thus, a flow need not necessarily receive service in each inner round. If the Pre-order DRR scheduler was serving flows in an exact round robin manner as in Nested DRR then, in the worst-case, it may be possible that none of the active flows will be able to transmit a packet in an inner round resulting in a work complexity of $O(n)$ or greater, where n represents the total number of active flows. The Classifier module in the Pre-order DRR scheduler avoids this large work complexity by classifying the packets into the p priority queues at the start of each outer round. This classification determines which inner rounds each flow will be served in and the scheduler does not need to query all the flows in a round robin order.

Note that the deficit count of a flow is updated at the end of each inner round using Eq. (5) irrespective of whether it receives service in that inner round. From Eq. (5), the service received by flow i in inner round (s, u) is,

$$Served_i(s, u) = \frac{Q_i}{p} + DC_i(s, u - 1) - DC_i(s, u) \quad (7)$$

Definition 2. Let $Sent_i(s, u)$ represent the total service received by flow i since the start of the s -th outer round until the time instant when the scheduler finishes serving the packets in the priority queue PQ_u .

$Sent_i(s, u)$ is computed as follows:

$$Sent_i(s, u) = \sum_{w=1}^{w=u} Served_i(s, w)$$

Substituting for $Served_i(s, w)$ from Eq. (7) in the above, we have,

$$Sent_i(s, u) = \left(\frac{u}{p}\right)Q_i + DC_i(s - 1) - DC_i(s, u) \quad (8)$$

$Sent_i(s, u)$ will be positive only if the sum of $(u/p)Q_i$ and $DC_i(s - 1)$ is greater than or equal to the size of the packet at the head of flow i . If this condition is not satisfied then it implies that flow i has not received any service in the first u inner rounds. However, each flow is guaranteed to receive service during at least one inner round within each outer round.

Definition 3. Define $Sent_i(s)$ as the total service received by flow i outer round s .

Note that, $Sent_i(s)$ is equal to $Sent_i(s, p)$. Therefore, substituting $u = p$ in Eq. (8), we get,

$$Sent_i(s) = Q_i + DC_i(s - 1) - DC_i(s) \quad (9)$$

4. Latency analysis of Pre-order DRR

In deriving an upper bound on the latency of Pre-order DRR, we use the concept of Latency-Rate (\mathcal{LR}) servers first proposed in Ref. [7]. The following definitions lead to a formal definition of latency for guaranteed-rate schedulers. The reader is referred to Ref. [7] for a more detailed discussion.

Definition 4. An active period of a flow is defined as the maximal interval of time during which at least one packet of the flow is either awaiting service or is in service.

Definition 5. A busy period of a flow is defined as the maximal time interval during which the flow is active if it is served at exactly its reserved rate.

Since the busy period of a flow assumes that a flow is served exactly at its reserved rate, it depends only on the reserved rate and the traffic arrival pattern of the flow. However, an active period of a flow reflects the actual behavior of the scheduler where the instantaneous service offered to the flow varies according to the number of active flows.

Definition 6. Let $Sent_i(t_1, t_2)$ be defined as the total service received by flow i during the time interval (t_1, t_2) .

Note that this notation is identical to the one used in Definition 2. Hence the reader should interpret $Sent_i(\beta, \gamma)$ based on whether β and γ represent two time instant or (β, γ) denotes an inner round in the execution of the Pre-order DRR scheduler.

Let the time instant α_i be the start of a busy period for flow i . Let $t > \alpha_i$ be such that flow i is continuously busy during the time interval (α_i, t) . Let $S_i(\alpha_i, t)$ be the number of bits belonging to packets in flow i that arrive after time α_i and are scheduled during the time interval (α_i, t) . Note that, during this time interval the scheduler may still be serving packets from a previous busy period, and hence $S_i(\alpha_i, t)$ is not necessarily the same as $Sent_i(\alpha_i, t)$.

Definition 7. The latency of a flow is defined as the minimum non-negative constant Θ_i that satisfies the following for all possible busy periods of the flow,

$$S_i(\alpha_i, t) \geq \max\{0, \rho_i(t - \alpha_i - \Theta_i)\} \quad (10)$$

As defined in Ref. [7], a scheduler which satisfies Eq. (10) for some non-negative constant value of Θ_i is said to belong to the class of Latency Rate (\mathcal{LR}) servers.

In practice, however it is easier to analyze scheduling algorithms based on the active period of a flow. Let flow i become active at time instant τ_i . Also let $t > \tau_i$ be some time instant such that the flow is continuously active during the time interval (τ_i, t) . Let Θ_i' be the smallest non-negative number such that the following equation is satisfied for all t .

$$Sent_i(\alpha_i, t) \geq \max\{0, \rho_i(t - \alpha_i - \Theta_i')\} \quad (11)$$

Even though (τ_i, t) may not be a continuously busy period for flow i , it has been proved in Ref. [7], that the latency as defined by Eq. (10) is bounded by Θ_i' . This allows us to determine the latency bound of a scheduler by considering only the flow active periods.

Theorem 1. The Pre-order DRR scheduler belongs to the class of \mathcal{LR} servers, with an upper bound on the latency Θ_i for flow i given by,

$$\Theta_i \leq \frac{1}{r} \left\{ \frac{(W - w_i)Q_{\min}}{p} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\} \quad (12)$$

where n is the total number of active flows, p represents the number of priority queues, W is the sum of the weights of all the flows and r denotes the transmission rate of the output link.

Proof. Since the latency of an \mathcal{LR} server can be estimated based on its behavior in the flow active periods, we will prove the theorem by showing that,

$$\Theta_i' \leq \frac{1}{r} \left\{ \frac{(W - w_i)Q_{\min}}{p} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$$

Let τ_i be the time instant when flow i becomes active. To prove the statement of the theorem we must consider an active period (τ_i, t) of flow i . We then obtain the lower bound on the total service received by flow i during the time interval under consideration. Lastly, we express the lower bound in the form of Eq. (10) to derive the latency bound. In Ref. [10] it has been proved that to obtain a tight upper

bound on the latency of the Elastic Round Robin scheduler [11,12], we need to consider only those active periods (τ_i, t) which satisfy the following two requirements:

1. τ_i coincides with the start of a service opportunity of some flow.
2. Time instant t belongs to a subset of all possible time instants at which the scheduler begins serving flow i .

It can be easily verified that these two conditions are applicable for proving the upper bound on the latency of the Pre-order DRR scheduler. Let $\tau_i^{(e,f)}$ be the time instant marking the start of the service of flow i when flow i is at the head of priority queue PQ_f in round e . In other words, this time instant represents the start of the service opportunity of flow i in inner round (e, f) . Note that $\tau_i^{(e,f)}$ belongs to the set of time instants when the scheduler begins serving flow i . Therefore, in order to determine the latency bound of the Pre-order DRR we need to only consider time intervals $(\tau_i, \tau_i^{(e,f)})$ for all (e, f) in which flow i receives service.

The first step toward analyzing the latency bound involves choosing a suitable time interval $(\tau_i, \tau_i^{(e,f)})$ such that the size of this time interval is the maximum possible. Note that the time instant τ_i may or may not coincide with the start of a new outer round. Let k_0 be the outer round which is in progress at time instant τ_i or which starts exactly at time instant τ_i . In either case, flow i will receive an opportunity to transmit Q_i worth of data in the k_0 -th round. Let the time instant t_h mark the start of the outer round $(k_0 + h)$. Consider the case when τ_i does not coincide with the time instant t_0 , the start of outer round k_0 , i.e., $\tau_i > t_0$. In this case, the time interval (t_0, τ_i) will be excluded from the time interval under consideration. On the other hand, when τ_i coincides with t_0 , the size of the time interval $(\tau_i, \tau_i^{(e,f)})$ is maximal. We, therefore, assume that the τ_i coincides with the start of the k_0 -th outer round. Fig. 1 illustrates the time interval under consideration assuming that (e, f) is equal to $(k_0 + k, v)$. Note that in Fig. 1, OR(a) represents the a -th outer round and $IR(a, b)$ denotes the inner round (a, b) in the execution of the Pre-order DRR scheduler. The time interval under consideration, $(\tau_i, \tau_i^{(k_0+k,v)})$, can be split into two sub-intervals:

- (1) (τ_i, t_k) : This sub-interval includes k outer rounds of execution of the Pre-order DRR scheduler starting at outer round k_0 . Consider the time interval (t_h, t_{h+1}) when outer round $(k_0 + h)$ is in progress. Summing Eq. (9) over all n flows,

$$t_{h+1} - t_h = \frac{W}{r} Q_{\min} + \frac{1}{r} \sum_{j=1}^n \{DC_j(k_0 + h - 1) - DC_j(k_0 + h)\} \quad (13)$$

Summing the above over k rounds beginning with round k_0 ,

$$t_k - \tau_i = \frac{W}{r} (kQ_{\min}) + \frac{1}{r} \sum_{j=1}^n \{DC_j(k_0 - 1) - DC_j(k_0 + k - 1)\} \quad (14)$$

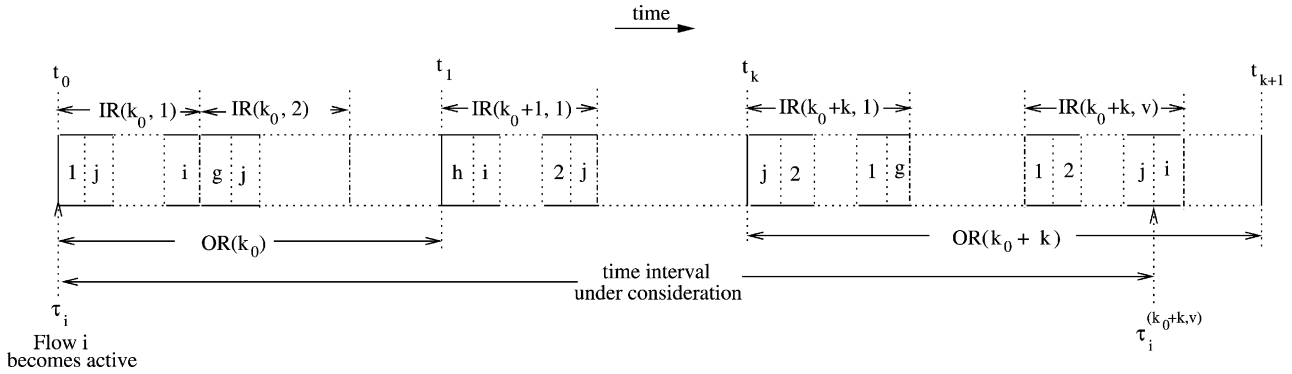


Fig. 1. An illustration of the time interval under consideration.

(2) $(t_k, \tau_i^{(k_0+k, v)})$: This sub-interval includes the part of the (k_0+k) -th round prior to the start of the service of flow i when it is at the head of priority queue PQ_v . In the worst-case, flow i will be the last flow to receive service among all the flows which may be present in priority queue PQ_v . In this case, during the sub-interval under consideration, the service received by flow i equals $Sent_i(k_0+k, v-1)$ whereas the service received by each flow j among the other $(n-1)$ flows equals $Sent_j(k_0+k, v)$. Note that if v equals 1 then flow i does not receive service in this sub-interval. Summing $Sent_i(k_0+k, v-1)$ and $Sent_j(k_0+k, v)$ for each flow j such that $1 \leq j \leq n, j \neq i$ and using Eq. (8), we have,

$$\begin{aligned} \tau_i^{(k_0+k, v)} - t_k &= \frac{1}{r} \sum_{\substack{j=1 \\ j \neq i}}^n \left(\frac{v}{p}\right) w_j Q_{\min} + \frac{1}{r} \left(\frac{v-1}{p}\right) w_i Q_{\min} \\ &+ \frac{1}{r} \sum_{\substack{j=1 \\ j \neq i}}^n (DC_j(k_0+k-1) - DC_j(k_0+k, v)) \\ &+ \frac{1}{r} (DC_i(k_0+k-1) - DC_i(k_0+k, v-1)) \end{aligned} \quad (15)$$

Combining Eqs. (14) and (15), we have,

$$\begin{aligned} \tau_i^{(k_0+k, v)} - \tau_i &= \frac{W}{r} (kQ_{\min}) + \frac{1}{r} \sum_{\substack{j=1 \\ j \neq i}}^n \left(\frac{v}{p}\right) w_j Q_{\min} \\ &+ \frac{1}{r} \left(\frac{v-1}{p}\right) w_i Q_{\min} + \frac{1}{r} \sum_{\substack{j=1 \\ j \neq i}}^n (DC_j(k_0-1) \\ &- DC_j(k_0+k, v)) + \frac{1}{r} (DC_i(k_0-1) \\ &- DC_i(k_0+k, v-1)) \end{aligned} \quad (16)$$

Now, since flow i becomes active at the start of outer round k_0 , its deficit count at the start of the k_0 -th outer round, $DC_i(k_0-1)$ is equal to zero. Using this fact and the bounds

on the deficit count from Eqs. (2) and (6) in Eq. (16), we have,

$$\begin{aligned} \tau_i^{(k_0+k, v)} - \tau_i &\leq \frac{W}{r} (kQ_{\min}) + \frac{1}{r} \sum_{\substack{j=1 \\ j \neq i}}^n \left(\frac{v}{p}\right) w_j Q_{\min} \\ &+ \frac{1}{r} \left(\frac{v-1}{p}\right) w_i Q_{\min} + \frac{(n-1)(m-1)}{r} \\ &- \frac{1}{r} DC_i(k_0+k, v-1) \end{aligned}$$

Solving for k ,

$$\begin{aligned} k &\geq (\tau_i^{(k_0+k, v)} - \tau_i) \frac{r}{WQ_{\min}} - \frac{r}{W} \sum_{\substack{j=1 \\ j \neq i}}^n \left(\frac{v}{p}\right) w_j \\ &- \frac{r}{W} \left(\frac{v-1}{p}\right) w_i - \frac{1}{WQ_{\min}} (n-1)(m-1) \\ &+ \frac{1}{WQ_{\min}} DC_i(k_0+k, v-1) \end{aligned} \quad (17)$$

Note that the total data transmitted by flow i during the time interval under consideration can be expressed as the following summation.

$$Sent_i(\tau_i, \tau_i^{(k, v)}) = Sent_i(\tau_i, t_k) + Sent_i(t_k, \tau_i^{(k, v)}) \quad (18)$$

As explained earlier, $Sent_i(t_k, \tau_i^{(k, v)})$ is the same as $Sent_i(k, v-1)$. $Sent_i(\tau_i, t_k)$ can be obtained by summing Eq. (9) over k outer rounds starting at outer round k_0 . Substituting the result of this summation and Eq. (8) in Eq. (18) and using the fact that the deficit count of a newly active flow is equal to zero, we have,

$$\begin{aligned} Sent_i(\tau_i, \tau_i^{(k_0+k, v)}) \\ = w_i (kQ_{\min}) + \left(\frac{v-1}{p}\right) w_i Q_{\min} - DC_i(k_0+k, v-1) \end{aligned} \quad (19)$$

Using Eq. (17) to substitute for k in Eq. (19), we get,

$$\begin{aligned}
& Sent_i(\tau_i, \tau_i^{(k_0+k,v)}) \\
& \geq \frac{w_i r}{W} (\tau_i^{(k_0+k,v)} - \tau_i) - \frac{w_i}{W} \left(\frac{v}{p} \right) (W - w_i) Q_{\min} \\
& \quad - \frac{w_i}{W} \left(\frac{v-1}{p} \right) w_i Q_{\min} - \frac{w_i}{W} (n-1)(m-1) \\
& \quad + \frac{w_i}{W} DC_i(k_0+k, v-1) + \left(\frac{v-1}{p} \right) w_i Q_{\min} \\
& \quad - DC_i(k_0+k, v-1) \tag{20}
\end{aligned}$$

Now, since the reserved rates are proportional to the weights assigned to the flows as given by Eq. (1), and since the sum of the reserved rates is no more than the link rate r , we have,

$$\rho_i \leq \frac{w_i}{W} r \tag{21}$$

Using Eq. (21) in Eq. (20), and simplifying we get,

$$\begin{aligned}
& Sent_i(\tau_i, \tau_i^{(k_0+k,v)}) \\
& \geq \rho_i (\tau_i^{(k_0+k,v)} - \tau_i) - \frac{\rho_i}{r} \left(\frac{v}{p} \right) (W - w_i) Q_{\min} \\
& \quad - \frac{\rho_i}{r} \left(\frac{v-1}{p} \right) (W - w_i) Q_{\min} - \frac{\rho_i}{r} (n-1)(m-1) \\
& \quad - \frac{\rho_i}{r} DC_i(k_0+k, v-1) \left(\frac{W}{w_i} - 1 \right)
\end{aligned}$$

Simplifying further, and noting that the latency bound reaches the upper bound when $DC_i(k_0+k, v-1)$ equals $(m-1)$,

$$\begin{aligned}
& Sent_i(\tau_i, \tau_i^{(k_0+k,v)}) \geq \\
& \quad \times \max \left\{ 0, \rho_i \left(\tau_i^{(k_0+k,v)} - \tau_i - \frac{1}{r} \left(\frac{(W - w_i) Q_{\min}}{p} \right. \right. \right. \\
& \quad \left. \left. \left. + (m-1) \left(\frac{W}{w_i} + n - 2 \right) \right) \right) \right\} \tag{22}
\end{aligned}$$

As discussed earlier, flow i will experience its worst latency during an interval $(\tau_i, \tau_i^{(k_0+k,v)})$ for some inner round (k_0+k, v) . Therefore, from Eq. (22), the statement of the theorem is proved. \square

We now proceed to show that the latency bound given by Theorem 1 is tight by illustrating a case when the bound is actually achieved. Let \mathbf{F} represent the set of all n flows. Assume that flow i becomes active at a certain time instant τ_i which also coincides with the start of certain outer round k_0 . Since the arrival of a packet into the empty buffer of a flow signals the start of a busy period of the flow, τ_i is also the start of its busy period. Assume that for any time instant $t, t > \tau_i$, a total of n flows, including flow i , are active. Also, assume that the summation of the reserved rates of all the n flows is equal to the transmission rate of the output link,

r . Therefore, we have, $\rho_i = (w_i/W)r$. Since flow i became active at time τ_i , its deficit count at the start of outer round k_0 is 0. Let the deficit count of all the other $(n-1)$ flows be equal to the maximum value of $(m-1)$. Using Eqs. (6) and (7), it is seen that the maximum service received by a flow j during an inner round, S_j^{\max} is given by,

$$S_j^{\max} = \frac{w_j Q_{\min}}{p} + (m-1) \tag{23}$$

On a similar note, the minimum service received by flow j during an inner round, S_j^{\min} , provided it is present in the priority queue being served, is given by,

$$S_j^{\min} = \frac{w_j Q_{\min}}{p} - (m-1) \tag{24}$$

Fig. 2(a) illustrates a part of the input traffic present in the queues of the n flows at the start of outer round k_0 . Fig. 2(b) shows how the *Classifier* module of the Pre-order DRR scheduler classifies these packets into the priority queues using Eq. (4). From Fig. 2(b) it can be seen that, except for flow i , all the other $(n-1)$ flows have packets classified into the highest priority queue PQ_1 . Prior to the service of the first packet of flow i , each flow $j, j \in \mathbf{F}, j \neq i$, transmits S_j^{\max} worth of data. Hence the cumulative delay until flow i receives service, X , is given by,

$$X = \sum_{\substack{j \in \mathbf{F} \\ j \neq i}} \frac{S_j^{\max}}{r}$$

Substituting for S_j^{\max} from Eq. (23), we have,

$$X = \frac{1}{r} \left(\frac{(W - w_i) Q_{\min}}{p} + (n-1)(m-1) \right) \tag{25}$$

Also the total flow i data that is served from PQ_2 equals S_i^{\min} .

Even though X represents the time for which flow i has to wait until it starts receiving service, Eq. (10) does not hold true if we substitute X as Θ_i . This is because in time interval $(\tau_i, \tau_i + X)$ flow i has not yet started receiving at its guaranteed rate. We assume that the latency, Θ_i is given by,

$$\Theta_i = X + Y \tag{26}$$

A plot of the service received by flow i against time is illustrated in Fig. 3. In order to determine the value of Y we shall consider the time interval $(\tau_i, \tau_i^{(k_0,2)})$ which satisfies the aforementioned requirements for deriving a tight upper bound on the latency. Referring to Fig. 3, we have,

$$Y + \frac{S_i^{\min}}{\rho_i} = \frac{1}{r} \sum_{\substack{j \in \mathbf{F} \\ j \neq i}} w_j Q_{\min} + \frac{S_i^{\min}}{r}$$

Substituting for S_i^{\min} from Eq. (24), we have,

$$Y + \frac{\frac{w_i Q_{\min}}{p}}{\rho_i} - \frac{m-1}{\rho_i} = \frac{\frac{W Q_{\min}}{p}}{r} - \frac{m-1}{r}$$

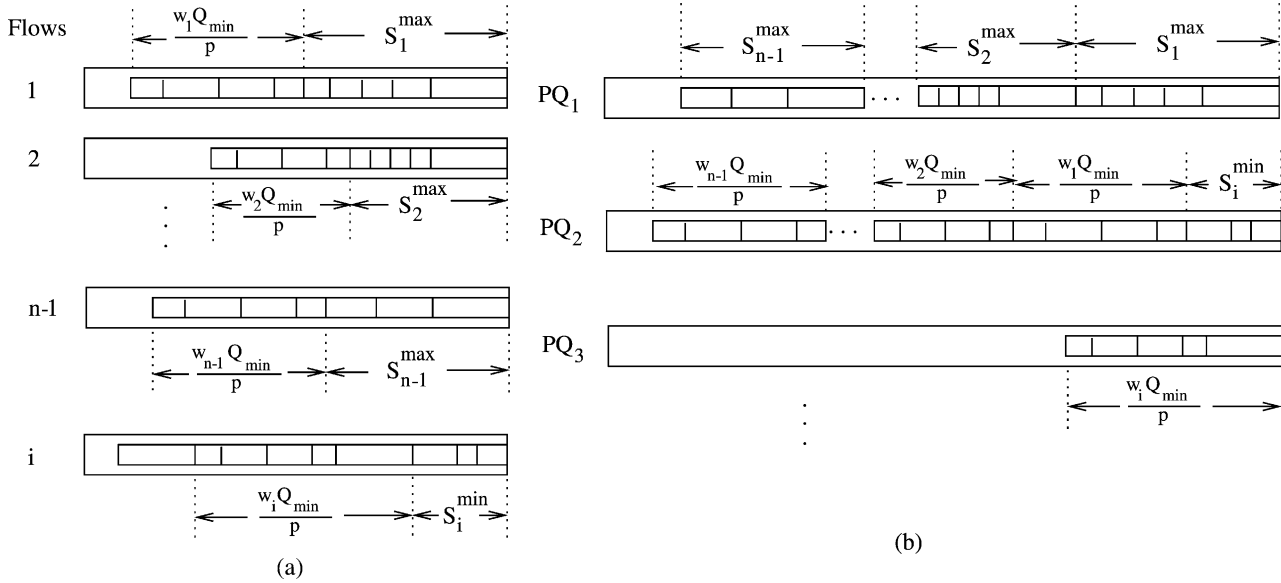


Fig. 2. (a) Input pattern; (b) packet Classification in the priority queues.

Now, since $\rho_i = (w_i/W)r$, simplifying further, we have,

$$Y = \frac{(m-1)}{r} \left(\frac{W}{w_i} - 1 \right) \quad (27)$$

Substituting for X and Y from Eqs. (25) and (27) in Eq. (26), it can be readily verified that the latency bound is exactly met.

4.1. A comparative study of the new latency bound

In this sub-section, we present a brief but detailed comparison of the latency bound of Pre-order DRR derived in Theorem 1 with the latency bound derived in Ref. [1]. Let Θ_i^{new} represent the former and let Θ_i^{old} denote the latter.

We also compare our latency bound with the latency bound of other efficient scheduling disciplines such as DRR [6], Elastic Round Robin (ERR) [12] and Nested DRR [8]. The latency bound of DRR and ERR are derived in Refs. [10,13], respectively. Let $\Theta_i^{\text{dr}}r$ and Θ_i^{err} represent the latency bounds of DRR and ERR. The expression for the upper

bound on the latency of Nested DRR, as proved in Ref. [8], is extremely complex and hence does not allow for an easy comparison with the other schedulers under consideration. Hence, in order to gain a quick understanding of the differences in the latency bounds of Nested DRR and Pre-order DRR, in our comparison we include the latency bound of Nested DRR at two boundary conditions. In the first case, we consider the latency bound of a flow whose reserved rate is much lower than that of the other flows sharing the same output link ($\rho_i \ll \rho_j, \forall j \in n, j \neq i$). In the second case, we consider the opposite end of the spectrum, i.e. the latency bound of a flow whose reserved rate is much greater than the other flows multiplexed on the same link ($\rho_i \gg \rho_j, \forall j \in n, j \neq i$). The former latency bound is represented as $\Theta_i^{\text{ndrr-low}}$ indicating a low-rate flow whereas the latter bound is represented as $\Theta_i^{\text{ndrr-high}}$. In Ref. [8], an expression for $\Theta_i^{\text{ndrr-low}}$ has been derived and it has also been shown that $\Theta_i^{\text{ndrr-high}}$ is marginally lower than $\Theta_i^{\text{dr}}r$. For simplicity we assume that $\Theta_i^{\text{ndrr-high}}$ is equal to $\Theta_i^{\text{dr}}r$.

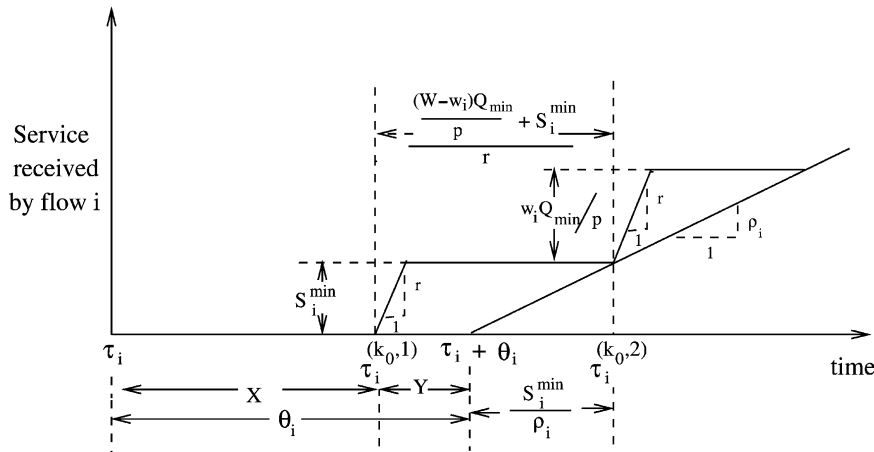


Fig. 3. Plot of service received by flow i .

Table 1
Comparison of latency bounds

Scheduler	Latency bound
Θ_i^{old}	$\frac{1}{r} \left\{ \frac{(W - w_i)Q_{\min}}{p} + Q_{\min} \left(2W - w_i + \frac{w_i}{p} \right) \right\}$
Θ_i^{new}	$\frac{1}{r} \left\{ \frac{(W - w_i)Q_{\min}}{p} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$
Θ_i^{drr}	$\frac{1}{r} \left\{ (W - w_i)Q_{\min} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$
Θ_i^{err}	$\frac{1}{r} \{ (W - w_i)m + (m - 1)(n - 1) \}$
$\Theta_i^{\text{ndrr-low}}$	$\frac{1}{r} \left\{ (n - 1)Q_{\min} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$
$\Theta_i^{\text{ndrr-high}}$	$\frac{1}{r} \left\{ (W - w_i)Q_{\min} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$

Table 1 summarizes all these latency bounds. Note that F , the size of a DRR frame, is equal to the sum of the quanta of all the active flows and is given by WQ_{\min} . Θ_i^{old} in Table 1 is obtained by substituting for F in the latency bound derived in Ref. [1]. Note that the latency bound of DRR stated here is tighter than the one proved in Ref. [14]. Our proof may be found in Ref. [13]. All the latency bounds in Ref. Table 1 are represented as a summation of two terms. Note that the first terms of Θ_i^{old} and Θ_i^{new} are identical and that the second term of Θ_i^{new} is equal to the corresponding term of Θ_i^{drr} . Recall that $Q_{\min} \geq M$ and that M . Therefore, it can be easily verified

that Θ_i^{new} is less than Θ_i^{old} , showing that the latency bound of Pre-order DRR as proved in this paper is a tighter bound. Also note that Θ_i^{old} is unable to distinguish between m and Q_{\min} . Note that in most networks including the Internet, the average packet size is much smaller than the maximum possible size [15,16] and therefore, $m \ll M$, resulting in $m \ll Q_{\min}$. Thus, in situations with small numbers of flows and where $m \ll M$, Θ_i^{new} can be much lower than Θ_i^{old} .

Comparing Θ_i^{drr} with Θ_i^{new} , it is seen that the first term of Θ_i^{new} is smaller than the corresponding term in Θ_i^{drr} by a factor of p . Thus, the latency bound of Pre-order DRR scheduler is significantly lower than that of DRR. It can be similarly verified that for the most part, latency bound of Pre-order DRR is lower than that of Nested DRR and ERR.

In order that the reader can fully appreciate the difference between the new and the old latency bounds, we provide a comparison of these two latency bounds of Pre-order DRR within the context of a practical example. Θ_i^{drr} , Θ_i^{err} and Θ_i^{ndrr} are also included to illustrate the improvement in latency achieved by Pre-order DRR. Note that in this comparison we use the actual latency bound for Nested DRR as proved in Ref. [8]. Let us assume that a total of 100 flows are multiplexed onto an output link whose transmission rate, r is 150 Mbps. Assume that M is equal to 576 bytes and let Q_{\min} be equal to M . Also assume that ρ_{\min} is equal to 0.1 Mbps and that the output link is completely utilized, i.e. $\sum_{i=1}^n \rho_i = r$. Note that this implies that the sum of all the weights is $150/0.1 = 1500$. Let the number of queues in the priority queue module of the Pre-order DRR scheduler, p , be equal to 10. We compare the latency bounds, Θ_i^{old} , Θ_i^{new} , Θ_i^{drr} , Θ_i^{err} and Θ_i^{ndrr} for flow i as a function of its

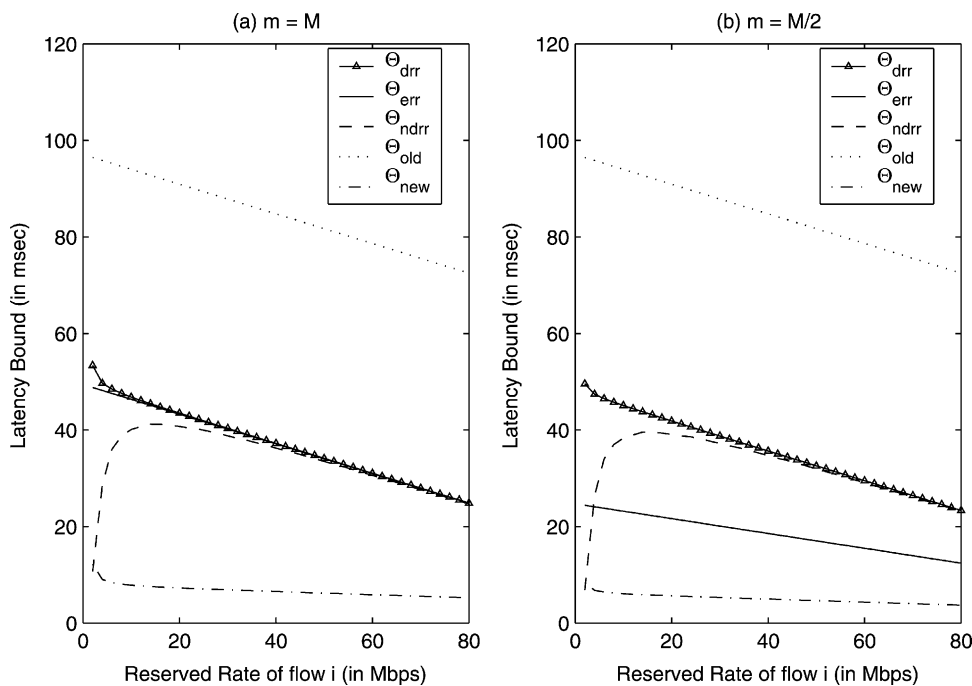


Fig. 4. Comparison of the latency bounds.

reserved rate, ρ_i , for two values of m : (a) $m = M$, (b) $m = M/2$. Fig. 4 illustrates a plot of these latency bounds of flow i for both values of m . Note that the expressions for the latencies of all the schedulers under consideration depend on the sum of the weights of all the flows but not on the distribution of the weights among all the flows other than flow i . Therefore, the weights of the flows other than flow i are not discussed in the context of this illustration. From Fig. 4, it can be seen that Θ_i^{old} is such a loose bound that it is in fact even greater than Θ_i^{dr} . On the other hand Θ_i^{new} is a tight bound, as proved earlier in this section.

5. Fairness analysis

The fairness of scheduling algorithms is most commonly judged by the *Relative Fairness Bound (RFB)*, a popular metric first proposed in Ref. [5] and later used in many other works. The RFB captures the maximum difference in the normalized service received by any two backlogged flows over all possible intervals of time. The RFB of the ideally fair GPS scheduler [2,3], of course, is 0. Certain other measures, such as the *Absolute Fairness Bound (AFB)* [17], are related to the RFB [18].

Table 2 lists the RFBs of the $O(1)$ round-robin schedulers. It is readily seen that Pre-order DRR has a lower RFB than both DRR and Nested DRR. On the other hand, comparing the RFBs of ERR and Pre-order DRR does not lead to an obvious conclusion. In addition, measures such as the RFB are based on the worst-case performance, which is just one aspect of the fairness achieved by a packet scheduler. For example, a scheduler that rarely reaches the upper bound of the fairness measure will achieve the same measure of fairness as another scheduler that frequently or almost always operates at the same upper bound. Therefore, we also need an *instantaneous* measure of fairness that captures the fairness achieved by the scheduler at *any* given instant of time.

To measure the fairness at any instant of time, we also need to consider situations in real applications. The RFB is defined under the assumption that queues are continuously backlogged in the interval of interest. Such an assumption is rarely true in real networks. In networks with real traffic, flow states can change frequently from active state to idle state, or vice versa. However, existing measures of fairness have not taken this factor into account. To effectively guide the design of a fair scheduler, a fairness measure should also

be able to capture the performance under the situation where flows change their states unpredictably.

A recently proposed measure of fairness described in Ref. [9] addresses these issues. There are two components to this measure: one of how to handle real traffic where flows are not always backlogged and the other of how to measure inequality in service received by the flows. We review these components briefly in this section; a detailed treatment and a more extensive rationale behind the approach may be found in Ref. [9].

5.1. Handling a newly backlogged flow

In order to evaluate the fairness of a scheduler in its treatment of a newly backlogged flow, we need to first define the ideally fair way of doing this. We begin with an examination of the ideal but unimplementable GPS scheduling discipline.

Let $B(t)$ represent the set of backlogged flows at time t . Also assume that the system starts at time $t = 0$.

Definition 8. Let $V(t)$ represent the virtual time function [3, 14] (also known as the system potential) at time t . The virtual time is used to track the progress of the GPS scheduler and is computed as follows:

$$V'(t) = \frac{dV(t)}{dt} = \left(\sum_{i \in B(t)} w_i \right)^{-1} \quad (28)$$

Hence, the service received by a backlogged flow under the GPS server in the time interval $(0, t)$ is given by $w_i V'(t)$. Intuitively, the virtual time represents the ideal fair normalized service that each flow should have received by time t .

Let us now consider a set of n flows served by a scheduling policy P . Consider a case in which the n flows have been backlogged since time t_1 . One of these flows, flow i , changes its status from being backlogged to idle at time $t_2 > t_1$ and later becomes backlogged again at time $t_3 > t_2$. In order to accurately and meaningfully compare the service received by all the flows at time instants after t_3 , it is necessary to assign an appropriate value of the normalized service received by flow i until t_3 so that the comparison is over the entire time interval (t_1, t_3) . As mentioned earlier, a newly backlogged flow should neither be favored nor be punished for its idle period in the interval (t_2, t_3) . Therefore, based on the discussion of the virtual time, the service received by flow i at time t should be $w_i V(t)$. However, if flow i has already received more service than the above amount of service before time t_2 while it was backlogged, then the total assumed service should be $\text{Sent}_i(0, t_2)$. This is because a flow that receives excess service should not be able to become idle and then immediately become backlogged again without being disadvantaged later for

Table 2
RFBs of efficient schedulers

Scheduler	RFB
DRR	$M + 2m$
ERR	$3m$
Nested DRR	$M + 2m$
Pre-order DRR	$(2M/p) + 2m$

the excess service it received earlier. These concepts and similar arguments have also been made in [5,7,9,19].

In evaluating the fairness of a specific scheduler it is necessary to keep track of the amount of service allocated by the above method. We borrow the method used in Ref. [9] where a per-flow state known as the *session utility* is defined for this purpose. This variable is independent of the scheduling discipline used by the scheduler and is defined as a function of time. Let $u_i(t)$ represent the session utility for flow i at time instant t . Assume that the system starts at time 0. During the period (t_1, t_2) that a flow is continuously backlogged, its session utility is updated as follows:

$$u_i(t_2) = u_i(t_1) + \frac{Sent_i(t_1, t_2)}{w_i} \tag{29}$$

We now discuss how to update the session utility of a flow that just becomes backlogged. Let flow i become newly backlogged or backlogged again at time t . Let $B(t-)$ represent the set of flows that are backlogged just prior to the time that flow i becomes backlogged. Our goal in assigning a session utility value to flow i at time t is to ensure that the comparison between session utilities of all the flows is being made as though the flows have all been backlogged for the same length of time. Accordingly flow i is assigned the following session utility value:

$$u_i(t) = \max\{u_i(t-), V(t)\} \tag{30}$$

With the above definition of the session utility, a newly backlogged flow can be treated as if it had been backlogged for the same length of time as all other flows. Therefore, with a measure which is based on session utility, it is possible to capture the fairness of a scheduler in its treatment of flows that are not always continuously backlogged.

5.2. The Gini index

Various measures of inequality have been used in the field of economics for several decades in the study of social wealth distribution and many other economic issues of interest [20]. Some of these methods are related to the theory of majorization used in mathematics as a measure of inequality [21]. This theory has occasionally found use in research in computer networks in the fairness analysis of protocols [22]. The fairness measure proposed in Ref. [9] and adopted in this paper borrows from a related measure of inequality developed in the field of economics based on the concept of the Lorenz curve and *Gini index* [20].

Consider the problem of measuring the inequality among k quantities, $g_1 \leq g_2 \leq \dots \leq g_k$. Define $d_0 = 0$, and $d_i = d_{i-1} + g_i$, for $1 \leq i \leq k$. Now, a plot of d_i against i is a concave curve, known as the *Lorenz curve* [23], as shown in Fig. 5(a). Note that if there is perfect equality in these k quantities, the Lorenz curve will be a straight line starting from the origin. The Gini index measures the area between the Lorenz curve and this straight line, and thus measures the inequality amongst the k quantities [20].

In our case, we wish to measure the inequality in the session utilities of the backlogged flows at any given instant of time. The Gini index in our case, therefore, is the area between the Lorenz curve of the actual normalized service received and the Lorenz curve corresponding to the ideally fair GPS scheduler.

When the sum of the k quantities is the same as the sum in the case of perfect equality, the Lorenz curve always lies below the straight line corresponding to the Lorenz curve of the ideal equal case, as shown in Fig. 5(a). However, the sum of the session utilities with a real scheduler is almost never exactly identical to the sum of the session utilities

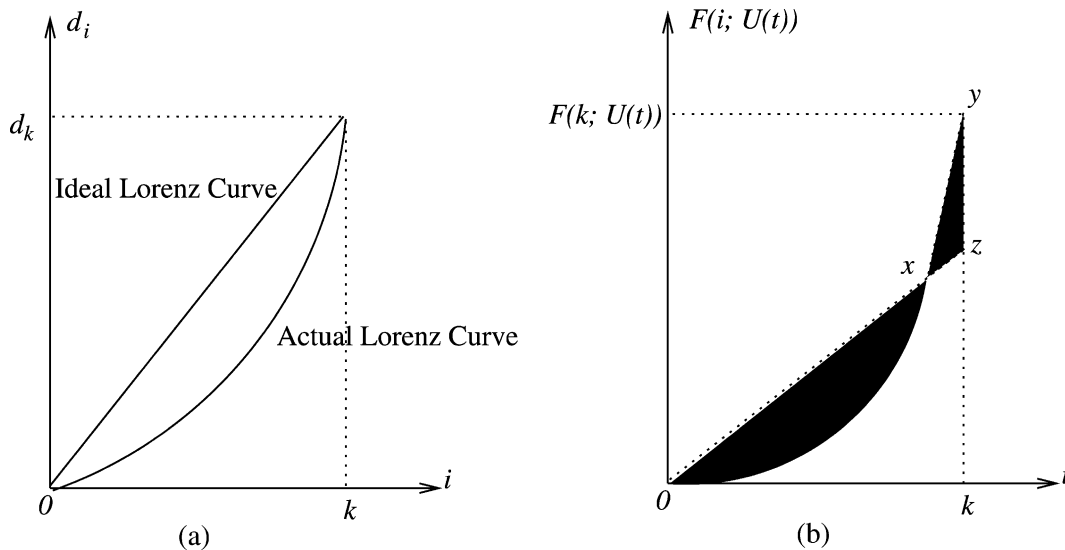


Fig. 5. An illustration of the Lorenz curve and Gini index in the measure of inequalities among: (a) income distribution; (b) session utilities in a packet scheduler.

Table 3
Settings for traffic sources from router traces

Source	1	2	3	4	5
Router Abbr. ^a	ANL	APN	BUF	MEM	TXS
Interface	OC3	OC3	OC3	OC3	OC3
L_{\min} (bytes)	28	29	28	32	32
L_{\max} (bytes)	9180	1500	1560	4470	9180
r_{avg} (10^6 Bps)	0.63	1.4	1.45	0.39	2.1
Weight (w_i)	1.6	3.5	3.7	1	5.5
Link capacity	6×10^6 Bps				
Total time	50 s				

^a The long names of routers are: Argonne National Laboratory (ANL), APAN (APN), University of Buffalo (BUF), University of Memphis (MEM) and Rice University (TXS).

with the ideally fair GPS scheduler. Note that, in a work-conserving scheduler, only the sum of the total service delivered is identical to that in the ideally fair GPS scheduler; the sum of the session utilities is not identical to that in the GPS system. In the Lorenz curve for a work-conserving scheduler, when the sum of the k quantities is not the same as the sum in the case of perfect equality as with the GPS scheduler, a portion of the curve for the actual scheduler will lie below and another portion will lie above the straight line Lorenz curve for the GPS scheduler. This is illustrated in Fig. 5(b). The sum of the shaded areas in the figure is the Gini index.

The computation of the Gini index is described formally as follows:

Definition 9. Let $\mathbf{U}(t)$ represent the set of the session utilities of the flows at time instant t when served by a real scheduler and let $\mathbf{G}(t)$ denote a similar set which is obtained when the flows are served with the ideal GPS scheduler. Let $u_{c_1}, u_{c_2}, \dots, u_{c_k}$ be the elements of the set $\mathbf{U}(t)$, such that $u_{c_1} \leq u_{c_2} \leq \dots \leq u_{c_k}$. The Lorenz Curve of the set of session utilities $\mathbf{U}(t)$ is the function $F(i; \mathbf{U}(t))$, given by,

$$F(i; \mathbf{U}(t)) = \sum_{j=1}^i u_{c_j}, \quad 0 \leq i \leq k$$

The Gini index over the k elements in $\mathbf{U}(t)$ is computed as:

$$\sum_{i=1}^k |F(i; \mathbf{U}(t)) - F(i; \mathbf{G}(t))| \quad (31)$$

As defined above, the closer the Lorenz curve of $\mathbf{U}(t)$ is to the curve of GPS, the smaller the Gini index is, and thus, the fairer the distribution of the session utilities. From the definition of the virtual time function, we know that the normalized service received by each flow at time t in the GPS system is equal to the virtual time, $V(t)$. Hence the Gini index can be computed as:

$$\sum_{i=1}^k |F(i; \mathbf{U}(t)) - F(i; V(t))| \quad (32)$$

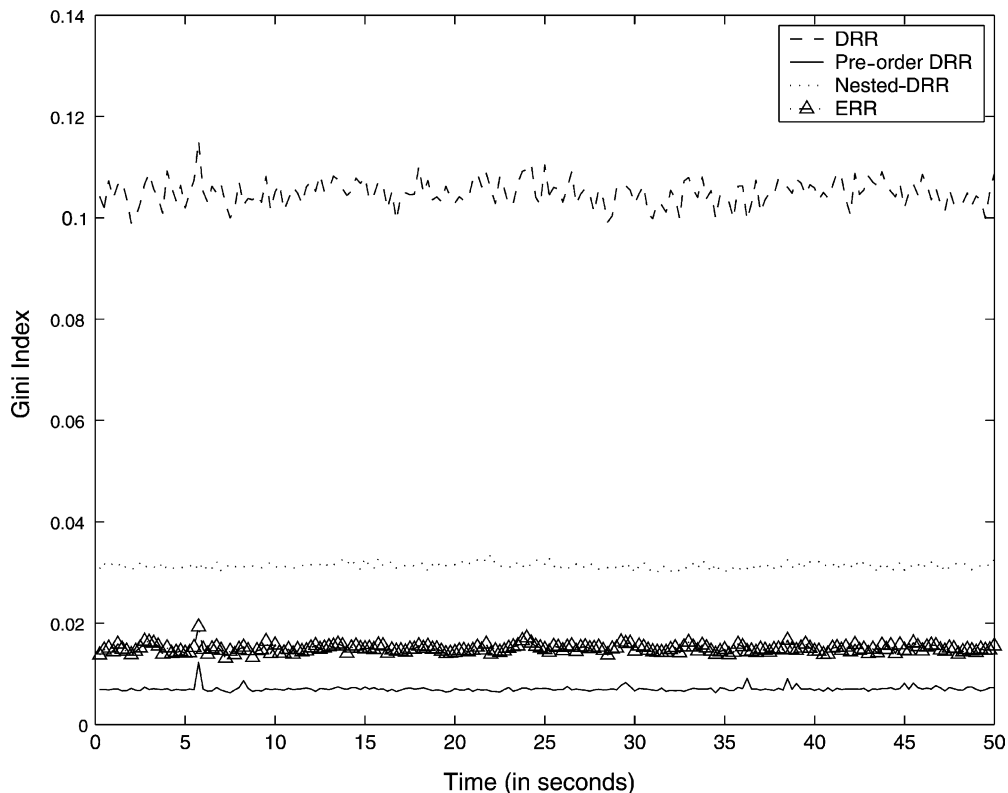


Fig. 6. Gini index of schedulers on backlogged queues with packet lengths from backbone router traffic traces.

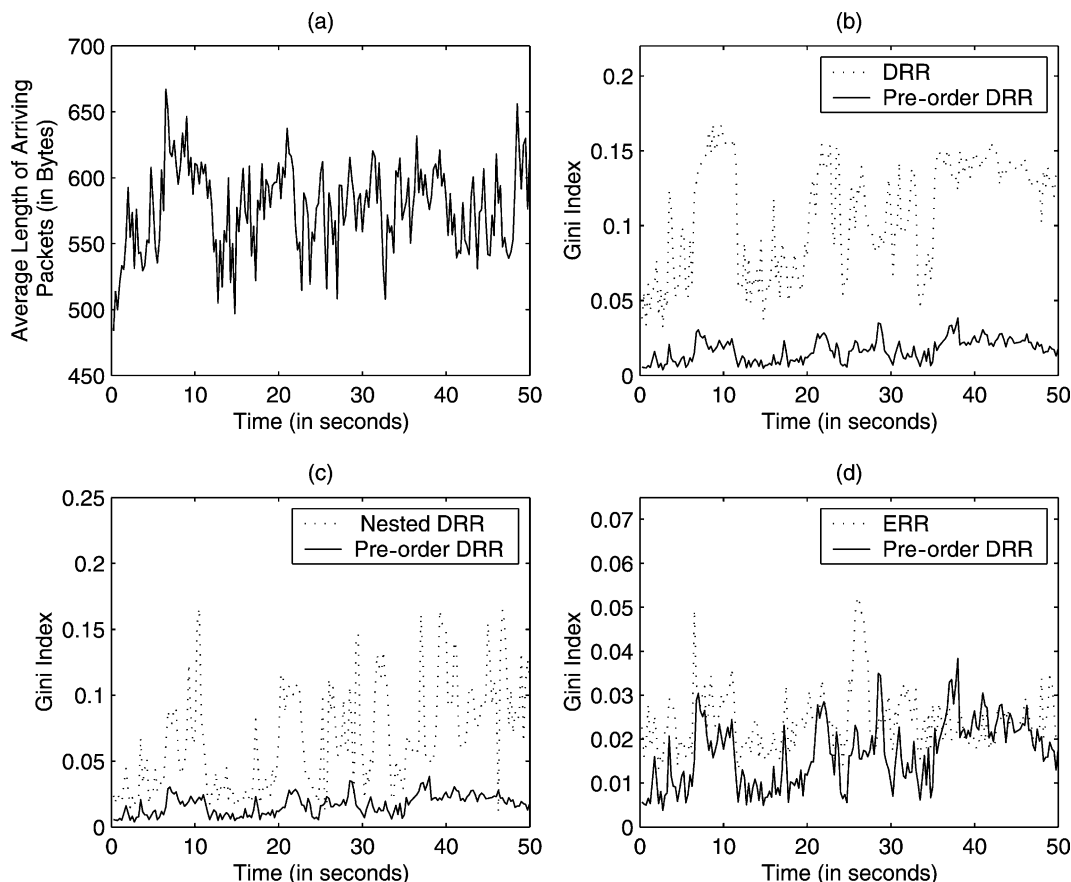


Fig. 7. Gini index of schedulers with backbone router traffic traces.

With the Gini index, the fairness of a scheduler can be evaluated at each instant during the execution of the scheduler. A comparison of schedulers based on their Gini indices allows us to determine which scheduler achieves better fairness than others at each instant of time.

5.3. Simulation results

In our simulation experiments, we compare the Gini index of Pre-order DRR with those of other efficient and fair schedulers such as DRR, ERR, and Nested DRR.

In our first set of experiments, the input queues are fed by backbone router traces. We used the traces provided by the National Laboratory for Applied Network Research [24]. Each input is fed by a router trace with a random starting time. Table 3 shows the settings for this set of input traffic. The flow weights are set based on the average rate of each flow. Here we set the weight of the slowest flow as 1, and weights of other flows are equal to the ratios of their average rate to the smallest rate.

We first extract the length of each packet from the router traces and simulate a scheduling system with continuously backlogged queues. Fig. 6 shows the Gini index at periodic instants of time for the schedulers under consideration. It can

be seen from Fig. 6 that Pre-order DRR displays the best fairness amongst all the schedulers.

In our next set of experiments, we allow that the flows are not always backlogged, while still using real router traces. Since we are more interested in the performance when the link is close to fully utilized, we set the link capacity such that the sum of average rates of all the flows is 99% of the link capacity. Fig. 7 shows the values of the Gini index observed for the four different schedulers. For the sake of clarity, we plot each scheduler's Gini index on a separate graph, with that of the Pre-order DRR scheduler plotted on each of the graphs. Fig. 7(a) shows the average length of arriving packets among all sessions during the simulation interval. Once again, we find that the Pre-order DRR scheduler displays a lower Gini index than any of the other schedulers at almost all the time instants. One may note that the ERR scheduler comes a close second.

6. Concluding remarks

Table 4 summarizes the work complexity with respect to the number of flows, the RFB, and the latency bounds of several guaranteed-rate scheduling disciplines that belong to the class of \mathcal{LR} servers. Note that the GPS

Table 4
A comparison between scheduling disciplines

Scheduling discipline	Complexity	Fairness (RFB)	Latency bound for flow i
GPS [3]	–	0	0
Weighted fair queueing [2]	$O(\log n)$	$O(n)$	$\frac{m}{r} + \frac{m}{\rho_i}$
SCFQ [5]	$O(\log n)$	$2m$	$\frac{(n-1)m}{r} + \frac{m}{\rho_i}$
Virtual clock [25]	$O(\log n)$	∞	$\frac{m}{r} + \frac{m}{\rho_i}$
Frame-based fair queueing [19]	$O(\log n)$	$2M + m$	$\frac{m}{r} + \frac{m}{\rho_i}$
DRR [6,13]	$O(1)$	$M + 2m$	$\frac{1}{r} \left\{ (W - w_i)M + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$
Elastic round robin [10,11]	$O(1)$	$3m$	$\frac{(W - w_i)m + (n - 1)(m - 1)}{r}$
Pre-order DRR [1]	$O(\log p)$	$\frac{2M}{p} + 2m$	$\frac{1}{r} \left\{ \frac{(W - w_i)M}{p} + (m - 1) \left(\frac{W}{w_i} + n - 2 \right) \right\}$

scheduler visits each active flow in a round-robin fashion and serves an infinitesimally small amount of data proportional to the reserved rate of the flow [3]. Using this fluid-flow model, the GPS scheduler is able to ensure that over any interval of time, however small, the normalized difference between the service received by any two active flows is exactly zero. The RFB of GPS, therefore, is zero. The latency of GPS is also zero since a newly active flow begins receiving service instantaneously at the guaranteed rate. Recall that GPS is an ideal but not an implementable scheduler. In this table, as in the rest of this paper, M is the size of the largest packet that may potentially arrive during the execution of a scheduling algorithm. Recall that m is the size of the largest packet that *actually* arrives during the execution of the scheduler. Typically, $M \gg m$, since in most networks including the Internet, the vast majority of the packets are of much smaller size than the maximum possible size of a packet [15,16]. The properties of all the scheduling disciplines in Table 4, except ERR and DRR, are derived in Ref. [14].

In summary, we have derived a tight upper bound on the latency of Pre-order DRR, demonstrating that it displays better performance characteristics than previously argued by its own authors [1]. Our approach to obtaining this latency bound uses a novel technique based on interpreting the Pre-order DRR bandwidth allocations as an instance of the Nested Deficit Round Robin (Nested DRR) discipline discussed in Ref. [8]. The results of this paper show that Pre-order DRR offers a better latency bound and better fairness characteristics than other scheduling disciplines of equivalent complexity.

The Pre-order DRR scheduler can be readily implemented by adding a priority queue module consisting of p queues and a *classifier* module to the original architecture of the DRR scheduler [1]. Note that, prior to the start of a round, the Pre-order DRR scheduler has to classify all the packets that can be transmitted by all the active flows in that round into the priority queues. As a result, this proposed implementation of the Pre-order DRR scheduler suffers from the following two drawbacks: (i) the priority queues need to store all the packets that can be transmitted in an entire round of service. This results in a significant overhead in the amount of buffers required as compared to the original DRR scheduler; (ii) More importantly, since the transmission cannot commence until the classification process is complete, the scheduler incurs a considerable delay prior to the start of each round.

However, these limitations can be easily overcome by introducing a few simple changes to the architecture of the Pre-order DRR scheduler. First, the priority queues could simply be used to store the flow identifiers as opposed to buffering the packets. In addition, the operation of the scheduler can be altered as described below. Let PQ_k represent the priority queue which is currently being serviced by the Pre-order DRR scheduler and let flow i be at the head of this queue. Following the service of the m -th packet from flow i , the priority queue PQ_z into which the flow is enqueued by the *Classifier*, provided it is still backlogged, is calculated as follows:

$$z = (k + \lfloor QA_i^m(s) \times p \rfloor) \bmod p \quad (33)$$

Recall that the *Quantum Availability*, $QA_i^m(s)$ is computed as described in Eq. (3). The above equation is also used to compute the priority queue for a newly active flow.

Once the Pre-order DRR scheduler has exhausted serving all the flows present in the current priority queue, PQ_k , it simply moves to the next non-empty priority queue in a circular order, i.e., following the service of queue PQ_p it resumes the service of queue PQ_1 . The proposed implementation eliminates all of the aforementioned drawbacks associated with the original architecture of the Pre-order DRR scheduler. Since the priority queues simply buffer the flow identifiers, the buffering requirement is low. Further, there is no delay associated with the start of each round since each flow is now added into its new priority queue immediately at the end of its current service opportunity. Finally, it also eliminates the need to maintain an *ActiveList* since, at any given instant of time, each active flow will be present in exactly one priority queue. Incorporation of these modifications makes the Pre-order DRR scheduler easily implementable in either hardware or software (Enun 1–11).

References

- [1] S. Tsao, Y. Lin, Pre-order deficit round robin: a new scheduling algorithm for packet-switched networks, *Computer Networks* 35 (2–3) (2001) 287–305.
- [2] A. Demers, S. Keshav, S. Shenker, Design and analysis of a fair queuing algorithm, *Proceedings of ACM SIGCOMM*, Austin (1989) 1–12.
- [3] A.K. Parekh, R.G. Gallager, A Generalized processor sharing approach to flow control—the single node case, *Proceedings of IEEE INFOCOM*, Florence, Italy (1992) 915–924.
- [4] J.C.R. Bennett, H. Zhang, WF²Q: Worst-case fair weighted fair queueing, *Proceedings of IEEE INFOCOM*, San Francisco, CA (1996) 120–128.
- [5] S.J. Golestani, A self-clocked fair queuing scheme for broadband applications, *Proceedings of IEEE INFOCOM*, Toronto, Canada (1994) 636–646.
- [6] M. Shreedhar, G. Varghese, Efficient fair queuing using deficit round-robin, *IEEE Transactions on Networking* 4 (3) (1996) 375–385.
- [7] D. Stiliadis, A. Verma, Latency-rate servers: A general model for analysis of traffic scheduling algorithms, *IEEE Transactions on Networking* 6 (3) (1996) 611–624.
- [8] S.S. Kanhere, H. Sethu, Fair, efficient and low-latency packet scheduling using nested deficit round robin, *Proceedings of the IEEE workshop on high performance Switching and Routing*, Dallas, TX (2001) 6–10.
- [9] H. Shi, H. Sethu, An evaluation of timestamp-based packet schedulers using a novel measure of instantaneous fairness, *Proceedings of the International Performance, Computing and Communications Conference*, Phoenix, AZ (2003) 443–450.
- [10] S.S. Kanhere, H. Sethu, Low-latency guaranteed-rate scheduling using elastic round robin, *Computer Communication* 25 (14) (2002) 1315–1322.
- [11] S.S. Kanhere, H. Sethu, A.B. Parekh, Fair and efficient packet scheduling in wormhole networks, *Proceedings of the International Parallel and Distributed Processing Symposium*, Cancun, Mexico (2000) 623–632.
- [12] S.S. Kanhere, H. Sethu, A.B. Parekh, Fair and efficient packet scheduling using elastic round robin, *IEEE Transactions on Parallel and Distributed Systems* 13 (3) (2002) 324–336.
- [13] S.S. Kanhere, H. Sethu, On the latency bound of deficit round robin, *Proceedings of the International Conference on Computer Communications and Networks*, Miami, FL (2002) 548–553.
- [14] D. Stiliadis, *Traffic Scheduling in Packet-Switched Networks: Analysis, Design and Implementation*, PhD Thesis, University of California, Santa Cruz, 1996.
- [15] K. Thompson, G.J. Miller, R. Wilder, Wide-area Internet traffic patterns and characteristics, *IEEE Network* 11 (6) (1997) 10–23.
- [16] I. Widjaja, A.I. Elwalid, Performance issues in vc-merge capable switches for multiprotocol label switching, *IEEE Journal on Selected Areas in Communications* 17 (6) (1999) 1178–1189.
- [17] S. Keshav, *An Engineering Approach to Computer Networks*, Addison-Wesley, Reading, MA, 1997.
- [18] Y. Zhou, H. Sethu, On the relationship between absolute and relative fairness bounds, *IEEE Communication Letters* 6 (1) (2002) 37–39.
- [19] D. Stiliadis, A. Varma, Efficient fair queuing algorithms for packet-switched networks, *IEEE Transactions on Networking* 6 (2) (1998) 175–185.
- [20] F.A. Cowell, *Measuring Inequalities: Techniques for the Social Sciences*, Wiley, New York, 1977.
- [21] A.W. Marshall, I. Olkin, *Inequalities: Theory of Majorization and its Applications*, Academic Press, New York, 1979.
- [22] A. Kumar, J. Kleinberg, Fairness measures for resource allocation, *Proceedings of the Symposium on Foundations of Computer Science* (2000) 75–78.
- [23] J.E. Stiglitz, *Economics*, W.W. Norton and Co, 1993.
- [24] National Laboratory for Applied Network Research, *Passive Measurement and Analysis*, <http://pma.nlanr.net/PMA/>.
- [25] L. Zhang, Virtual clock: a new traffic control algorithm for packet switching networks, *Proceedings of ACM SIGCOMM*, Philadelphia, PA (1990) 19–29.