

An evaluation of fair packet schedulers using a novel measure of instantaneous fairness

Hongyuan Shi, Harish Sethu*, Salil S. Kanhere

Department of Electrical and Computer Engineering, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA

Received 13 October 2003; revised 28 March 2005; accepted 5 April 2005

Available online 5 May 2005

Abstract

A number of emerging Internet applications such as video-conferencing and live multimedia broadcasts from Internet TV stations rely on scheduling algorithms in switches and routers to guarantee performance and an acceptable level of quality of service. Fairness in packet schedulers is an intuitively desirable property with practical value; fair schedulers such as weighted fair queueing are a critical component of quality-of-service mechanisms that seek to guarantee end-to-end delay bounds, and thus provide end-to-end service differentiation. Popular measures of the fairness achieved by packet schedulers are based on bounds, such as the relative fairness bound which captures the maximum possible difference between the normalized services received by any two flows. In this paper, we argue that such measures do not capture the actual fairness achieved at most instants of time, and therefore, do not represent a true measure of the ability of a scheduler to successfully deliver end-to-end quality for real-time applications. In this paper, we borrow from the field of economics and propose a new measure of fairness based on the Gini index. This measure captures the instantaneous fairness of a scheduler and, unlike other measures based on bounds, also captures the fairness of the scheduler in its handling of flows during idle periods. We use this measure on real gateway traffic traces and present a simulation-based evaluation of several well-known timestamp-based and frame-based schedulers. We also present a qualitative analysis of the phenomena underlying the observed results.

© 2005 Elsevier B.V. All rights reserved.

Keywords: Resource allocation; Fair scheduling; Fairness; Max–min fairness; WFQ

1. Introduction

A number of emerging Internet multimedia applications will rely on scheduling algorithms in switches and routers to guarantee performance and an acceptable level of quality of service. In a DiffServ framework with heterogeneous service classes, the scheduling algorithm achieves the desired quality of service for each class by determining the exact sequence in which packets should be transmitted. An intuitively desirable and also a practically important property of a packet scheduler is the fairness it achieves in the allocation of the bandwidth resources on the output link. A fair scheduler may also be used to provide strict delay guarantees required for interactive applications. Even with

the over-provisioning of resources such as is typical in the Internet core, fairness in scheduling is essential to protect flows from other misbehaving flows triggered by deliberate misuse or malfunctioning software on routers or end-systems. Fair scheduling becomes especially critical in access networks, within metropolitan area networks and in wireless networks where the resource capacity constraints tend to be significantly limiting to high-bandwidth multimedia applications today. Fair schedulers have now found widespread implementation in switches and Internet routers [1,2].

Fair schedulers attempt to allocate bandwidth resources by a certain allocation scheme based on a particular notion of fairness. The most commonly used notion of fairness is max–min fairness based on the intuition that no flow should receive more service than its demand and that no flow with an unsatisfied demand should receive less service than any other flow [3,4]. The generalized processor sharing (GPS) scheduler is proposed as an ideally fair but unimplementable scheduler that exactly achieves the goal of this notion of fairness [5]. During each infinitesimal interval of time,

* Corresponding author. Tel.: +1 215 895 5876; fax: +1 215 895 1695.

E-mail addresses: shi@ece.drexel.edu (H. Shi), sethu@ece.drexel.edu (H. Sethu), salil@ece.drexel.edu (S.S. Kanhere).

the GPS scheduler visits each backlogged flow once and schedules an infinitesimal amount of data proportional to the flow's weight for transmission over the output link.

Over the last decade, a number of different packet-by-packet scheduling algorithms have been proposed that seek to approximate the GPS scheduler. In the design of a fair packet scheduler, measures of fairness are defined as a useful analytical tool to evaluate a scheduler's performance. Among these measures, the most popular one is the relative fairness bound (RFB) [4,6], first proposed in [7] and later used in several other works [8–12]. The RFB captures the maximum possible difference between the normalized services received by any two backlogged flows, and therefore, serves as a measure of fairness. Since the RFB measures the worst-case unfairness, it cannot tell how likely it is that a scheduler is being fair at any given instant of time. For example, a scheduler that is almost always unfair in its allocation of bandwidth and another that is rarely unfair may both be evaluated to be equal in fairness by such a measure. In addition, the RFB cannot measure how fairly a scheduler distributes the available bandwidth amongst flows other than the extreme two used to compute the RFB. These aspects of fairness have significant relevance to end-to-end quality experienced by an application since they represent overall performance of the network and not just the worst-case performance. For example, in voice communications, the upper bound on the delay (related to the upper bound used in fairness measures) is not as significant as the delay experienced by packets most of the time.

An additional weakness of measures of fairness based on a bound is that the bounds are derived under the assumption that all flows are continuously backlogged during the period of interest. Real flows of traffic frequently go back and forth between a backlogged state to one in which they are not backlogged. An adequate measure of fairness should accommodate this reality. Almost all timestamp-based schedulers have proposed a principle based on which they determine the treatment a flow gets when it changes to a backlogged state. However, RFB and other measures cannot quantify the fairness of these principles and are unable to comparatively evaluate the fairness of a scheduler's technique of handling a newly backlogged flow.

In this paper, we analyze these deficiencies of currently used measures of fairness and propose a new measure, adapted from the measures of inequality used in the field of economics, to overcome these deficiencies. Our measure, called the *Gini index*, complements the information provided by traditional measures of fairness based on bounds. Together with these traditional measures, a more complete evaluation of a scheduler for real-time multimedia traffic becomes possible. In this paper, we provide such an evaluation of popular packet schedulers using real gateway traffic traces. We also provide a thorough qualitative analysis to uncover the underlying phenomena behind the observed results.

This paper is organized as follows. In Section 2, we discuss several fair packet schedulers which have been proposed in the research literature during the last decade. We also discuss several measures of fairness used to study the performance of packet schedulers, along with their unique limitations. Section 3 presents a new fairness measure based on requirements that emerge from the limitations of existing measures discussed in Section 2. Section 4 presents a simulation-based evaluation of popular schedulers using backbone router traces. This section also presents a qualitative analysis of the observed results. Section 5 concludes the paper.

2. Fairness in packet scheduling

2.1. Fair packet schedulers

Over the last decade, a number of different packet-by-packet scheduling algorithms have been proposed that seek to approximate the ideally fair GPS scheduler. The earliest such algorithm was *Weighted Fair Queueing* (WFQ) [5,13] which tries to emulate the GPS scheduler by timestamping each arriving packet with a finish number, the expected completion time of the packet if it were scheduled by the ideally fair GPS scheduler. The WFQ scheduler then serves the packets in increasing order of their finish numbers. Thus, the WFQ scheduler seeks to emulate GPS through preserving the same order in the packet transmissions as in GPS. A number of different variants of WFQ have been proposed which seek to either improve the accuracy or reduce the complexity in the computation of the finish number. *Self-Clocked Fair Queueing* (SCFQ) [7] uses the finish number of the packet currently being transmitted in the computation of finish numbers for arriving packets, and thus achieves an easier implementation and also, very good fairness. *Start-Time Fair Queueing* (SFQ) [8] is a variant of SCFQ which uses the starting time of the packet currently in service to compute the timestamp of the arriving packet. Certain other variants of WFQ use additional eligibility criteria in the determination of the next packet to transmit. For example, *Worst-Case Fair Weighted Fair Queueing* (WF²Q) [14] transmits the packet with the lowest finish number among those that would have already begun transmission under the GPS scheduler.

A different approach to the design of fair scheduling algorithms was proposed by Stiliadis and Varma based not on an explicit emulation of GPS through the use of finish numbers, but instead based on periodically re-calibrating a global variable indicating the progress of the scheduler. This reduces the complexity of timestamp computations and can be shown to permit the design of provably fair and computationally efficient schedulers [9]. A somewhat similar approach of timestamping flows instead of packets, leading to a similar level of computational efficiency, is used in time-shift scheduling [12]. In this scheme, each flow

is assigned an increasing timestamp and the packet chosen for transmission is from the flow with the least timestamp.

All the above schedulers belong to the class of sorted priority schedulers, which maintain a global variable such as the virtual time or the system potential. This variable is used to compute the sequence in which packet transmissions are scheduled. Sorted priority schedulers are often also known as *timestamp-based schedulers*. Another class of scheduling algorithms are those based on round robin or frame-based approaches [10,11] which do not achieve as good a fairness as most of the schedulers discussed above but which are significantly simpler to implement in both hardware and software. A well-known example is *Deficit Round Robin (DRR)* [10]. While serving flows in a round robin order, DRR eliminates the unfairness due to different packet lengths by maintaining a *deficit counter* for each queue as a record of past unfairness. The amount of service allocated to each flow is determined by the quantum of the queue and the deficit counter from the last service. A similar method was proposed in [15,16], later known as *Surplus Round Robin (SRR)*. Several variants of DRR have been proposed to further reduce the unfairness of scheduling decision, such as *Nested-DRR* [17] and *Pre-order DRR* [18]. All the above schedulers require the knowledge of the maximum packet length to ensure a low computation complexity. Another frame-based scheduler, *Elastic Round Robin (ERR)* [11], uses a method which does not need the maximum packet length to achieve fairness with low complexity. It adjusts the amount of service allocated to each flow based on the recent service history. ERR achieves better performance than other schedulers with a per-packet computational complexity of $O(1)$. Recently, a further improvement upon ERR was proposed in [19]. The new scheduling discipline, called *Prioritized Elastic Round Robin (PERR)*, borrowing the principle used in Pre-order DRR, re-orders the transmission sequence within each frame of ERR. It achieves a fairness close to that of timestamp-based schedulers with a lower processing complexity than timestamp-based schedulers.

2.2. Fairness measures

To study the fairness measure of a fair packet scheduler, we introduce the following notation. Consider a set of flows, $1 \leq i \leq n$, which share an output link of peak bandwidth rate R . Flow i is associated with a weight w_i . The smallest of the weights is denoted by w_{\min} , and the largest by w_{\max} . Denote the sum of all weights as W . The total service received by flow i over time interval (t_1, t_2) under a given scheduling discipline P is denoted by $S_i^P(t_1, t_2)$. If the scheduling discipline under consideration is GPS, this total service is denoted by $S_i^G(t_1, t_2)$.

One can measure the fairness of a real scheduler by measuring how close its bandwidth allocation is to the ideally fair GPS scheduler. An early measure of fairness achieved by a scheduler, used in the works by Parekh and Gallager [5] and also by Bennett and Zhang [14], captures

the upper bound on the difference in the service received with the real scheduler and that with GPS since the beginning of a backlogged period for any given flow. More formally, the measure is defined as follows:

$$\max_{\forall \tau \geq 0} (S_i^G(0, \tau) - S_i^P(0, \tau))$$

This measure, however, does not use normalized service, i.e. service received by a flow normalized by its weight, and therefore, in the above form, is not an adequate measure of fairness in packet schedulers which achieve weighted fair allocation.

The fairness of scheduling algorithms is most commonly judged by the *relative fairness bound (RFB)* [4,6], a popular measure of fairness first used in [7] and later used in several other works [8–12]. The RFB captures the maximum possible difference between the normalized services received by any two backlogged flows, and therefore, serves as a measure of fairness. The RFB of the ideally fair GPS scheduler, of course, is 0. We now provide a formal definition of the RFB.

Definition 1. Let (t_1, t_2) be an interval of time during which all flows under consideration are continuously backlogged while being served by the scheduling policy, P . The relative fairness measured with respect to a pair of flows (i, j) over time interval (t_1, t_2) , denoted by $\text{RF}_{(i,j)}(t_1, t_2)$, is defined as

$$\text{RF}_{(i,j)}(t_1, t_2) = \left| \frac{S_i^P(t_1, t_2)}{w_i} - \frac{S_j^P(t_1, t_2)}{w_j} \right| \quad (1)$$

The relative fairness with respect to a flow i over time interval (t_1, t_2) , denoted by $\text{RF}_i(t_1, t_2)$, is defined as

$$\text{RF}_i(t_1, t_2) = \max_j \text{RF}_{(i,j)}(t_1, t_2) \quad (2)$$

The relative fairness over time interval (t_1, t_2) , $\text{RF}(t_1, t_2)$, and the relative fairness bound (RFB) can now be defined as

$$\text{RF}(t_1, t_2) = \max_i \text{RF}_i(t_1, t_2) \quad (3)$$

$$\text{RFB} = \max_{\forall (t_1, t_2)} \text{RF}(t_1, t_2) \quad (4)$$

A related measure of fairness, called the *absolute fairness bound (AFB)*, captures the upper bound on the difference between the normalized service received by a flow under the scheduler P and that it would receive under the ideally fair GPS scheduler [4]. It can be shown that the absolute and relative fairness measures are related to each other by a simple relationship [6].

Another fairness measure, known as the *worst-case fair index*, is defined in [20]. It is used to analyze the WF^2Q scheduler in [14]. However, this measure is not used as widely as the RFB.

2.3. Requirements of fairness measures

In this section, we elaborate on the weaknesses of existing measures of fairness. The measures of fairness described above, including the RFB, are all based on the worst-case performance. However, the worst-case performance bound 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 not, even though it should, achieve a better measure of fairness than another scheduler that frequently or almost always operates at the same upper bound. Therefore, besides the worst-case bound, 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. 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. In fact, the way to handle a newly backlogged flow has an important impact on the overall fairness performance of a scheduler and should not be neglected in an evaluation under real traffic scenarios. Actually, the fairness of treating a newly backlogged flow has been absorbed into the design of most timestamp-based fair packet schedulers. 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.

In addition, fairness measures based on upper bounds on the difference in services received by each flow do not inform us of how a scheduler treats packets of one flow in comparison to those of another. Fairness, after all, is expected to be based on a comparison among the levels of service received by all the flows and not merely on the maximum difference in the normalized service received by flows. Fig. 1(a) and (b) illustrates an example where the bars represent the service received by flows under two different schedulers A and B, during a certain interval of time in

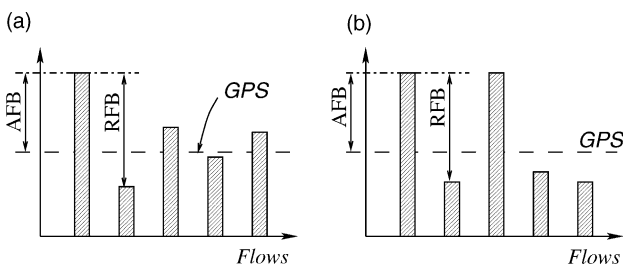


Fig. 1. An illustration of the difference in the disparity in service received while the upper bounds of the relative and absolute fairness measures are identical.

which all flows are backlogged. Assuming the weights associated with the flows are identical, the dashed line represents the level of service each of the flows would receive under the ideally fair GPS scheduler. One may observe from the figures that scheduler B leads to a greater disparity in the levels of service received by the flows since scheduler A allows more flows to achieve service close to the ideally fair level. If the absolute and relative fairness bounds are exactly reached in this interval of time, note that both schedulers would lead to the same values for the RFB and the AFB, even though, the levels of service received by flows under scheduler A are closer to each other than with scheduler B.

Thus, a measure of fairness should capture the overall behavior of the scheduler at all instants of time, including those time periods when flow states change dynamically. It also should capture the overall fairness of the allocation amongst all the flows and not just the extreme two flows. In the following, we propose a new metric to measure the fairness of a packet scheduler which can meet the above requirements.

3. A measure of instantaneous fairness

3.1. 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 so. Clearly, it is unreasonable to expect that a fair scheduler would allocate the same amount of total normalized service to a flow that is just backlogged as it would to another flow, which has been backlogged for a long time. Therefore, we claim that to fairly treat a newly backlogged flow is to neither favor it nor punish it for its idle period.

Consider a case in which N flows, including flows i and j , have been continuously backlogged since time 0 till t_1 . An ideally fair scheduler would allocate equal normalized service to each flow during this interval, as done in a GPS scheduler. Suppose flow i changes its status from being backlogged to idle at time $t_2 > t_1$ and later becomes backlogged again at time $t_3 > t_2$. As we claimed before, a newly backlogged flow should neither be favored nor be punished for its idle period in the interval (t_2, t_3) . Therefore, for purposes of comparisons between the services received by flows, we should assume that flow i at time t_3 has received service equal to:

$$w_i \frac{S_j^G(t_1, t_3)}{w_j} \quad (5)$$

However, if flow i has already received more than the above amount of service before time t_2 while it was backlogged, then total assumed service should be $S_i^G(t_1, t_2)$. 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 the excess service it received earlier. These concepts and similar arguments have also been made in [7,9,21].

In evaluating the fairness of a specific scheduler, we need to keep track of the amount of service allocated by the above method. Therefore, we define a state variable independent of the scheduling discipline used in a scheduler. This variable is called the *session utility*. It is defined for each flow as a function of time. Therefore, we denote it by $(u_i(t))$ for flow i .

Assume that the system starts at time 0. During the period (t_1, t_2) that a flow i is continuously backlogged, its session utility is updated as follows:

$$u_i(t_2) = u_i(t_1) + \frac{S_i(t_1, t_2)}{w_i}$$

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 . 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

$$u_i(t) = \max\{u_i(t-), m(t)\} \tag{6}$$

where $m(t)$ is the normalized service received on a continuously backlogged flow in the GPS reference scheduler, and the notation ‘ $t-$ ’ denotes the time instant just before time t .

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 based on the session utility, it is possible to capture the fairness of a specific scheduler in its treatment of flows that are not continuously backlogged from the beginning of the system.

3.2. 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 [22]. Some of these methods are related to the theory of majorization used in mathematics as a measure of inequality [23]. This theory has occasionally found use in research in computer networks in the fairness analysis of protocols [24]. Here we borrow the idea of a popular, mature and one of the least ambiguous measures of inequality developed in the field of economics based on the concept of the Lorenz curve and Gini index [22].

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* [25], as shown in

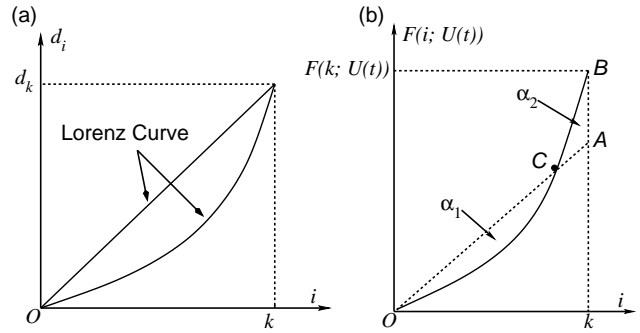


Fig. 2. An illustration of the Lorenz curve and Gini index in the measure of inequality amongst (a) income distribution, (b) session utilities in a packet scheduler.

Fig. 2(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 [22].

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 session utilities 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. 2(a). However, the sum of the session utilities is almost never exactly identical to the sum of the session utilities in the ideally fair GPS case. 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, which represents the sum of the *normalized service* delivered, is not identical to that in the GPS system. In 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 Lorenz 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. 2(b). The sum of the areas marked by α_1 and α_2 in the figure represent the Gini index. Instead of computing the actual area, the sum of the absolute differences at each point between the ideal and actual Lorenz curves capture a quantity proportional to the area. We describe the computation of the Gini index formally in the following.

Definition 2. Let $\mathbf{U}(t)$ denote the set of session utilities obtained with a real scheduler at time t , and let $\mathbf{G}(t)$ denote the same with the 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:

$$\sum_{i=1}^k |F(i; \mathbf{U}(t)) - F(i; \mathbf{G}(t))| \tag{7}$$

Using the same notation in the discussion of setting session utility, $m(t)$ represents the amount of normalized service a flow should receive in GPS system. The Gini index is computed as:

$$\sum_{i=1}^k |F(i; \mathbf{U}(t)) - F(i; m(t))| \tag{8}$$

As defined above, the closer the Lorenz curve of $\mathbf{U}(t)$ is from the curve of GPS, the smaller the Gini index is, and thus, the fairer is the distribution of normalized services. With the Gini index of session utilities in a packet scheduler, it is possible to tell which scheduler is fairer than others at any instant of time.

The fairness of a scheduler can be evaluated at each instant during the execution of the scheduler.

3.3. Why the Gini index?

In this section, we briefly present the relevance of using the Gini index to our context, the importance and advantages of it in comparison to other common statistical measures and the rationale behind defining the Gini index as in Section 3.2.

The Gini index, as used in the economics literature, can also be interpreted as the relative mean difference of a set of quantities. In other words, if we define the mean difference of g_1, g_2, \dots, g_k as

$$\Delta = \frac{1}{k^2} \sum_{i=1}^k \sum_{j=1}^k |g_i - g_j|$$

the Gini index is one-half of the relative mean difference which is the mean difference divided by the mean \bar{g} , i.e.

$$\text{Gini} = \frac{\Delta}{2\bar{g}} = \frac{1}{2k^2\bar{g}} \sum_{i=1}^k \sum_{j=1}^k |g_i - g_j| \tag{9}$$

As shown in the expression of the Gini index based on the relative mean difference, the Gini index incorporates the difference between any two quantities in the group. A similar measure might be the sum of the distances from the mean, divided by the mean. However, this metric gives more emphasis on extremely large or small quantities; it is difficult to tell whether the inequality within an income distribution is due to the difference amongst most quantities or due to the

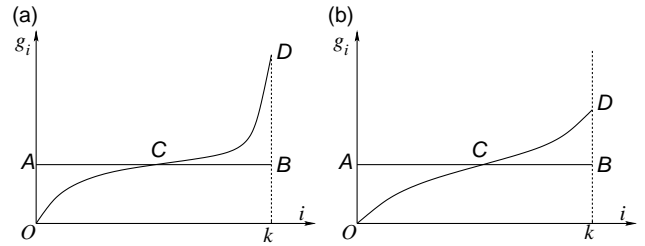


Fig. 3. An illustration of two distributions.

difference between quantities with extremely large or small values. This ambiguous situation is illustrated using examples in Fig. 3. Fig. 3(a) and (b) plots two groups of k quantities. In both graphs, curve OCD represents the value of k quantities and line AB represents the mean value of the same k quantities. The sum of the distance from the mean is nothing but the area between line AB and curve OCD . We can observe that both groups have the same mean value and the area between line AB and curve OCD is almost the same in both graphs. However, most quantities are close to each other and close to the mean value in Fig. 3(a) while most quantities are different from each other in Fig. 3(b). Therefore, the sum of the distances from the mean does not quite capture the differences that exist between the majority of the quantities.

Another common metric used by statisticians is variance or standard deviation. It has the same shortcoming as the sum of distances from the mean. The Gini index, on the other hand, captures the inequality or unfairness among a group of unequal quantities. It is for these reasons that we use it to measure the inequality among session utilities and therefore measure the instantaneous fairness in bandwidth allocation. However, the characteristics of session utilities require that we slightly modify the original definition of the Gini index. The following explains the modifications and the rationale behind it.

The first modification is about choosing the set of equal quantities. When we compute the Gini index for, say, the social income distribution, the straight line associated with the ideal equal distribution can be obtained by dividing the total income by the total number of persons. However, in a packet scheduler, we cannot use the same method since not all flows are continuously backlogged from the beginning of time in the system. The GPS virtual time records the normalized service a flow should receive in a GPS system, an ideal system, which allocates service equally among all backlogged flows. Therefore, we use the GPS virtual time as the mean of session utilities and the straight line is associated with the GPS virtual time, which is also the normalized service in the GPS reference system.

Similar to how the Gini index is used in economics literature, we define the fairness metric as the area between the Lorenz curves of the real scheduling system and the GPS reference system. Since the sum of normalized service in a real system may not equal that in a GPS system even though the sum of total service is equal in the two systems, the end

point of the Lorenz curve of a real system may not always meet the end point of the curve of the GPS system. Therefore, under certain conditions, the Lorenz curve of a packet scheduling system may appear similar to that in Fig. 2(b). This results in another modification of the Gini index. The Gini index in economics divides the area between the straight line (the Lorenz curve for an equal distribution) and the Lorenz curve for the distribution being measured by the total area under the straight line. For packet schedulers, we cannot directly use such a measure since the total area under the Lorenz curve of the GPS system can increase as time elapses, while the area between Lorenz curves from the GPS and the real system will not increase significantly if the real system is close to being fair. In order to avoid a misleading result because of the relative area reducing as the system operation continues, our definition of the Gini index for packet schedulers in Section 3.2 uses only the absolute area (as opposed to a ratio of the areas).

4. Evaluation using the Gini index

For the purposes of our evaluation using the Gini index, we focus on the following timestamp-based and frame-based packet schedulers:

- WFQ [5,13], for obvious historical reasons. Its RFB is derived in [26].
- WF²Q [14], because it achieves the best possible lag in comparison to the ideally fair fluid flow GPS scheduler, and has therefore been commonly believed to be the most fair packet scheduler. The RFB of WF²Q can be derived from the bounds of service lag and lead.
- Self-Clocked Fair Queueing (SCFQ) [7], as a representative scheduler among those with the best known relative fairness bound.
- Starting Potential Fair Queueing (SPFQ) [9], a representative scheduler based on the concept of rate-proportional servers introduced in [21]. Its RFB is slightly larger than SCFQ but less than WFQ.

- Time-Shift Scheduling (TSS) [21], since it has a similar RFB with SPFQ but uses flow-based timestamps as opposed to per-packet timestamps.
- Deficit Round Robin (DRR) [10], since it is one of the most popular frame-based schedulers, variants of which have found implementation in operating systems and routers [1,2].
- Elastic Round Robin (ERR) [11], since it boasts the best relative fairness bound among frame-based schedulers with O(1) per-packet complexity.
- Surplus Round Robin (SRR) [15,16]. It uses a similar discipline as DRR and has the same RFB as DRR.
- Nested-Deficit Round Robin (nested-DRR) [17]. It improves the fairness when flows carry significantly different weights, even though it has the same RFB as DRR.
- Pre-order Deficit Round Robin (Pre-order DRR) [18]. It combines priority queueing with DRR and thus achieves a good trade-off between fairness and work complexity.
- Prioritized Elastic Round Robin (PERR) [19]. This scheduler achieves a fairness close to that of timestamp-based schedulers although it has the same work complexity as Pre-order DRR.

Table 1 summarizes the RFBs of above schedulers. The left part of the table includes the RFBs of timestamp-based schedulers. Among these schedulers, SCFQ has the smallest RFB. The RFBs of SPFQ and TSS are slightly worse than SCFQ. WF²Q has a RFB worse than SCFQ, but better than WFQ. WFQ has the worst RFB among all timestamp-based schedulers under study. The right part of the table lists the RFBs of frame-based schedulers under study. Among DRR, ERR, SRR and nested-DRR, ERR has the lowest RFB, while the rest all have the same RFB. The RFB of PERR is smaller than that of Pre-order DRR.

Table 1 shows an evaluation of schedulers based on the RFBs. However, our evaluation using the Gini index does not show the same results as those indicated by the RFB measures.

Table 1
RFBs of the schedulers under study

Scheduler	RFB ^a	Scheduler	RFB ^a
WFQ	$3 \frac{l_{\max}}{w_{\min}}$	DRR, SRR	$\frac{L_{\max} + 2l_{\max}}{w_{\min}}$
WF ² Q	$C \leq \text{RFB}_{\text{WF}^2\text{Q}} \leq \text{RFB}_{\text{WFQ}}$	ERR	$3 \frac{l_{\max}}{w_{\min}}$
SCFQ	$2 \frac{l_{\max}}{w_{\min}}$	Nested-DRR	$\frac{L_{\max} + 2l_{\max}}{w_{\min}}$
SPFQ	$2 \frac{l_{\max}}{w_{\min}} + \frac{l_{\max}}{W}$	Pre-order DRR	$\frac{2L_{\max}p + 2l_{\max}}{w_{\min}}$
TSS	$2 \frac{l_{\max}}{w_{\min}} + \frac{l_{\max}}{W}$	PERR	$\frac{2L_{\max}p + 2l_{\max}}{w_{\min}}$

^a l_{\max} is the maximum length of a packet that actually arrives; L_{\max} is the maximum length of a packet that will potentially arrive during the execution of a scheduler; $C = (2 + (w_{\min}/W - w_{\min}))L_{\max}/w_{\min}$ is a constant derived from $\text{AFB}_{\text{WF}^2\text{Q}}$ using the relation in [6]; p is the number of priority queues in pre-order DRR and prioritized ERR.

Table 2
Settings for traffic sources from router traces

Source	1	2	3	4	5
Router abbr. ^a	ADV	ANL	APN	BUF	TXS
Interface	OC3	OC3	OC3	OC3	OC3
L_{\min} (bytes)	38	28	29	32	28
L_{\max} (bytes)	4470	9180	1500	1560	9180
r_{avg} (10^6 Bps)	0.56	0.63	1.4	1.45	2.1
Weight (w_i)	1	1.1	2.5	2.6	3.8
Link capacity	6.2×10^6 Bps				
Total time	50 s				

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

4.1. Simulation environment

In the simulation experiments, input queues are fed by real Internet gateway traces [27]. Each input is fed by a router trace with a random starting time. The settings in this set of experiments are listed in Table 2. The flow weights are set based on the observed 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.

4.2. Results assuming backlogged traffic

We first extract the length of each packet from the router traces and simulate a scheduling system with continuously backlogged queues. Fig. 4(a) shows the average length of packets in transmission during each period of 250 ms. It provides us a rough idea of the variations in packet lengths in the real trace. Throughout the simulation period, the average packet length ranges from 460 to 650 bytes.

Our observations show that TSS, SCFQ and SPFQ have Gini index values very close to that of WFQ even though their RFBs are different. This is because the timestamp of a new packet is only related to that of the packet at the tail of a queue when queues are all continuously backlogged and these four schedulers use similar methods of computing the timestamp based on the timestamp at the tail of a queue. The actions based on the timestamp in these schedulers are also similar. Even though the worst-case performance is quite different among them, the scheduling orders of SCFQ, SPFQ, TSS and WFQ are nearly the same throughout the simulation period. As a result, we choose to plot only the performances of WF²Q and WFQ in Fig. 4. Recall that the smaller the Gini index is, the better the fairness achieved by the scheduler. It is clearly seen from these graphs that WF²Q is more fair than WFQ.

Fig. 4(d) plots the Gini index of DRR, SRR, ERR, nested-DRR, Pre-order DRR and PERR. As is readily seen from the graph, PERR has the best fairness and is very close to that of WFQ. A more detailed plot of the Gini index of

PERR is shown in Fig. 4(e). Among the rest of the schedulers, Pre-order DRR comes the closest to PERR. Note that Pre-order DRR and PERR both have per packet work complexity of $O(\log p)$, where p is the number of priority queues used in their implementation. In this experiment, p is set to 10 for both schedulers.¹

A closer view of the Gini index of DRR and SRR is plotted in Fig. 4(f). The performance of SRR is very close to that of DRR. However, SRR achieves slightly better fairness than DRR throughout the simulate of this experiment.

Nested-DRR and ERR outperform SRR and DRR by a large margin. This is because, in both SRR and DRR, a pre-allocated *quantum* worth of transmission is scheduled from each flow during each service opportunity. To ensure an $O(1)$ work complexity, this quantum has to be greater than or equal to the size of the largest packet that may potentially arrive, L_{\max} . Irrespective of the actual size of the arriving packets, both these schedulers (on an average) serve a quantum's worth of data from each flow in each round robin service opportunity given to a flow. However, in reality, a large percentage of the arriving packets are much smaller than L_{\max} . As a result, these schedulers do not achieve good short-term fairness. Nested-DRR breaks the traditional round robin order and arranges the transmission in a nested round robin fashion. Thus, it has substantially better fairness properties in comparison to the DRR and SRR schedulers. The ERR scheduler reduces the unfairness of DRR and SRR by constantly adjusting the allowed average amount of transmissions per service opportunity. As a result, the allowance of a flow in ERR is much lower than the flow's quantum in DRR and SRR most of the time, resulting in improved fairness for the ERR scheduler. ERR also performs better than nested-DRR which uses the same method as DRR to set the quantum for each flow. Therefore, the ERR scheduler has the best fairness properties among all $O(1)$ scheduling disciplines in this evaluation.

Note that, even though ERR has the same RFB as WFQ, WFQ achieves much better instantaneous fairness than ERR according to the Gini index.

4.3. Results with real arrival characteristics

We now use real gateway traffic traces without forcing the assumption that traffic will always be backlogged. This evaluation is relevant to reality since flows will typically go back and forth between a backlogged state and a non-backlogged state and the schedulers may perform quite differently under these conditions. Since we are 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 flows is about 99% of the link capacity. Throughout the simulation in this set of experiments, the traffic load is

¹ One might notice that p is larger than the number of queues in the simulation. However, the value of p is not related to the number of flows. In reality, the number of flows is usually in the order of thousands, while p is likely to be a much smaller number ranging between 2 and 16.

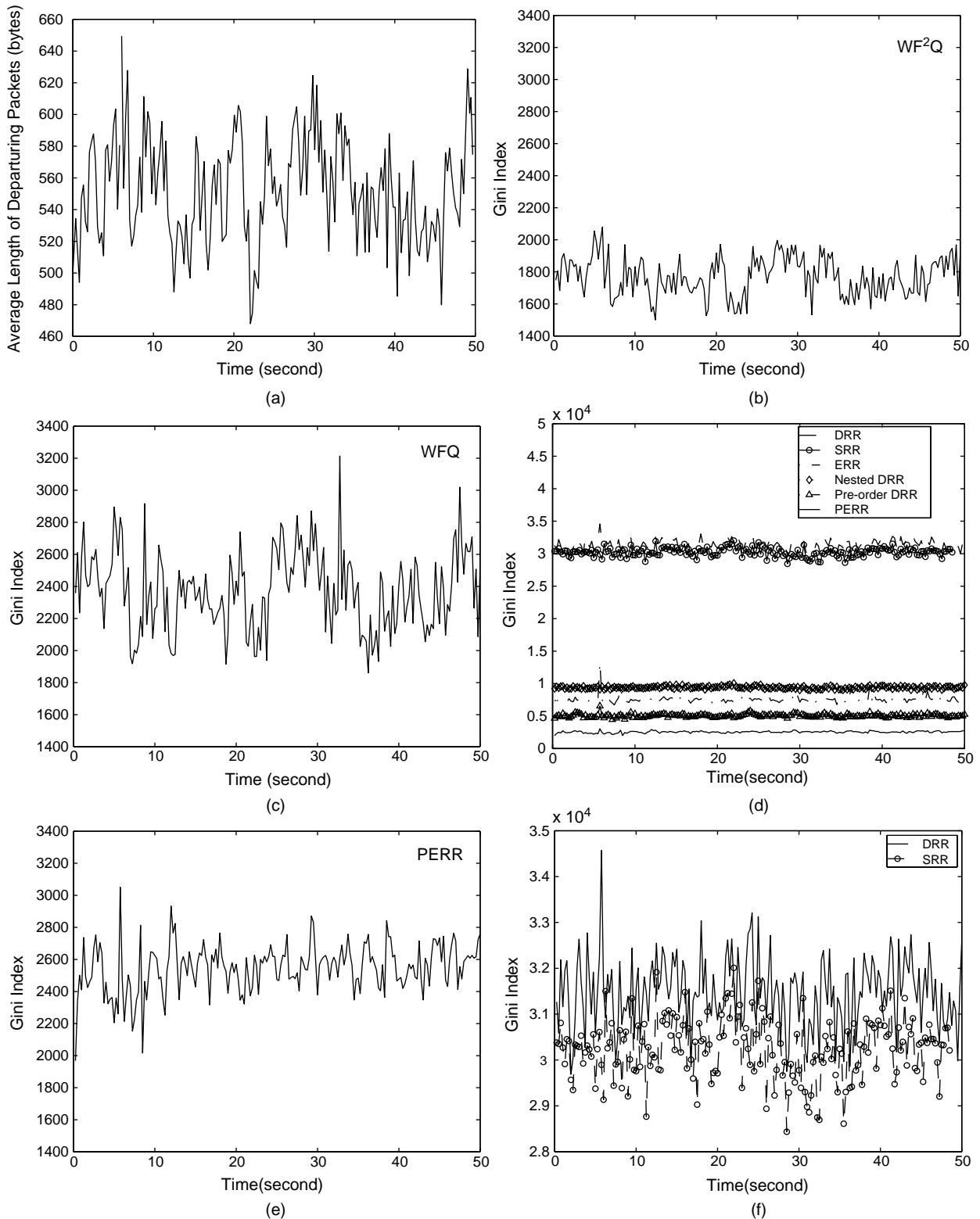


Fig. 4. Gini indices of schedulers on backlogged queues with packet lengths from backbone router traffic traces.

measured by the average rate over a periodic interval of 250 ms, and is plotted in Fig. 5. There are three overloaded periods approximately during the time intervals 5–10, 20–30 and 35–45 s.

4.3.1. Timestamp-based schedulers

Fig. 6 shows the values of the Gini index observed for the five timestamp-based schedulers. One frame-based scheduler, PERR, is also included here since

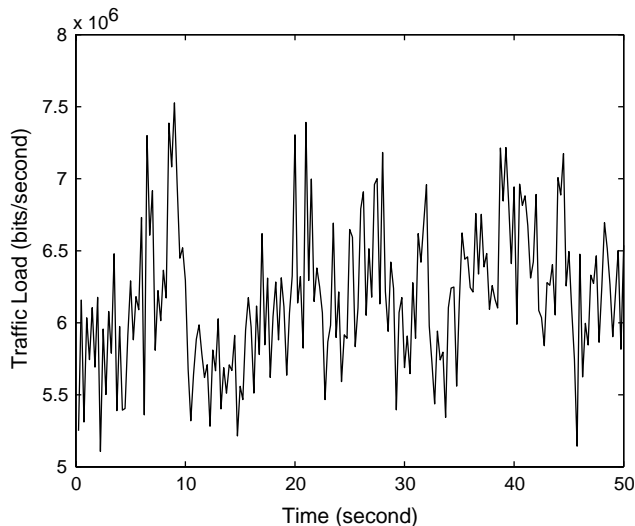


Fig. 5. Traffic load during the simulation with backbone router traffic traces.

its performance is very close to that of timestamp-based schedulers. For clarity in comparison, we plot each scheduler's Gini index distribution on a separate graph.

As is readily observed, WF²Q is the most fair scheduler among those studied here. Amongst the other schedulers, SPFQ, TSS and SCFQ have similar fairness performance, as shown in Fig. 6(c)–(e). However, except for a few short unfair intervals, SPFQ and TSS still exhibit slightly better fairness than SCFQ. Even though SCFQ has a smaller RFB than SPFQ and TSS, it generates several unfair transmission decisions during the bursts. This is because SCFQ uses the finishing time stamp of the packet currently being served as the start time for newly arrived packets in all sessions. This introduces a discrepancy in the computation of the timestamp in the flows other than the one currently being served. This discrepancy does not get corrected at a later time, leading to a less fair allocation. SPFQ and TSS solve the above problem by using the starting value corresponding to the packet currently being served and the time elapsed since the current transmission starts. One can see from Fig. 6(c) and (d) that the fairness achieved by these two schedulers is similar to each other and is slightly closer to WF²Q than SCFQ.

As shown in Fig. 6(b), WFQ is unfair during the burst periods. This result is in conformance with the RFB evaluation since WFQ has the worst RFB amongst timestamp-based schedulers under study. The Gini index of PERR is plotted in Fig. 6(f), using the scale of this group of graphs. The fairness property of PERR is very close to timestamp-based schedulers. In this set of experiments, it achieves equal or better fairness than all timestamp-based schedulers except WF²Q.

4.3.2. Frame-based schedulers

Here we compare the instantaneous fairness of DRR, ERR, SRR, nested-DRR, pre-order DRR and PERR.

The traffic sources are same as in Section 4.3.1. Both of these schedulers re-order traffic at each scheduling junction into p separate queues. The number of these queues, p , in both pre-order DRR and PERR is set to 10 in our experiments. Fig. 7(a)–(f) shows the fairness measured by the Gini index of the six schedulers. Similar to the results in Section 4.2, ERR exhibits better fairness than DRR, SRR and nested-DRR. With real traffic sources, nested-DRR performs closer to DRR and SRR than to ERR. This is due to the fact that frame-based schedulers do not treat a newly backlogged flow fairly. Since nested-DRR also uses a fixed value for the transmission allowance as in the case of DRR, the result is a large unfairness experienced by a newly backlogged flow.

Fig. 7(f) plots the Gini index of PERR again. In this graph, the value of the Gini index is the same as in Fig. 6(f), but the scale used is changed so as to enable comparison to other frame-based schedulers. As shown in Fig. 7(d)–(f), PERR achieves better fairness than Pre-order DRR. Recall that both schedulers use 10 queues into which the packets are re-ordered at each scheduling junction, and thus have the same work complexity in this experiment. Compared to ERR, PERR does improve the fairness to a great extent, while pre-order DRR achieves only slightly better fairness than ERR.

5. Concluding remarks

In this paper, we argue that existing measures of fairness do not accurately capture the actual fairness achieved at most instants of time, and therefore, do not represent a true measure of the ability of a scheduler to successfully deliver end-to-end quality for real-time applications. For example, the worst-case delay bound is not important to end-to-end quality in voice communications if such delay occurs only rarely; the delay that most packets experience is ultimately more critical to end-to-end quality. In this paper, we borrow from the field of economics and propose a new measure of fairness based on the *Gini index*. This measure captures the instantaneous fairness of a scheduler and, unlike other measures based on bounds, also captures the fairness of the scheduler in its handling of flows during idle periods. Both these characteristics of the new measure are important for the design of scheduling disciplines for multimedia traffic, since such traffic tends to have heavy-tailed distribution of packet lengths and highly irregular patterns of backlogged and idle periods. Using real gateway traffic traces, we present a simulation-based evaluation of several well-known timestamp-based and frame-based schedulers based on the Gini index. We also present a qualitative analysis of the phenomena underlying the observed results. Our study, in comparison to other measures in the networking literature, represents a more accurate evaluation of packet schedulers for end-to-end quality in real-time communications.

From the simulation-based evaluation using the Gini index, we observe that WF²Q achieves the best fairness

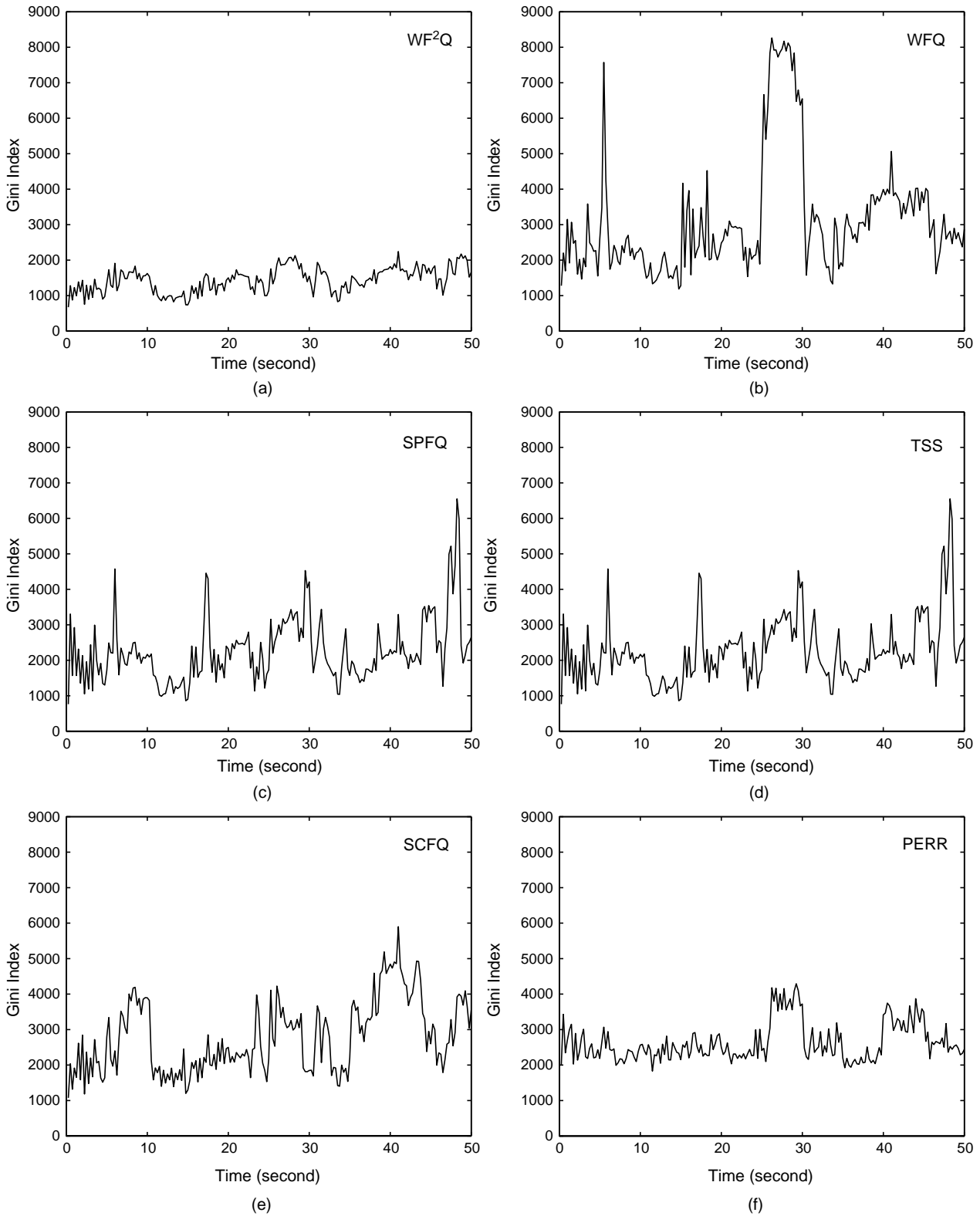


Fig. 6. Gini indices of timestamp-based schedulers and PERR with backbone router traffic traces.

amongst timestamp-based schedulers. ERR achieves the best fairness amongst frame-based schedulers of $O(1)$ per-packet work complexity. PERR, a recently proposed frame-based scheduler, achieves fairness that is very close

to that achieved by timestamp-based schedulers but with a significantly lower per-packet work complexity.

Our future research includes the development of a novel scheduler that incorporates, in its design, the goal of

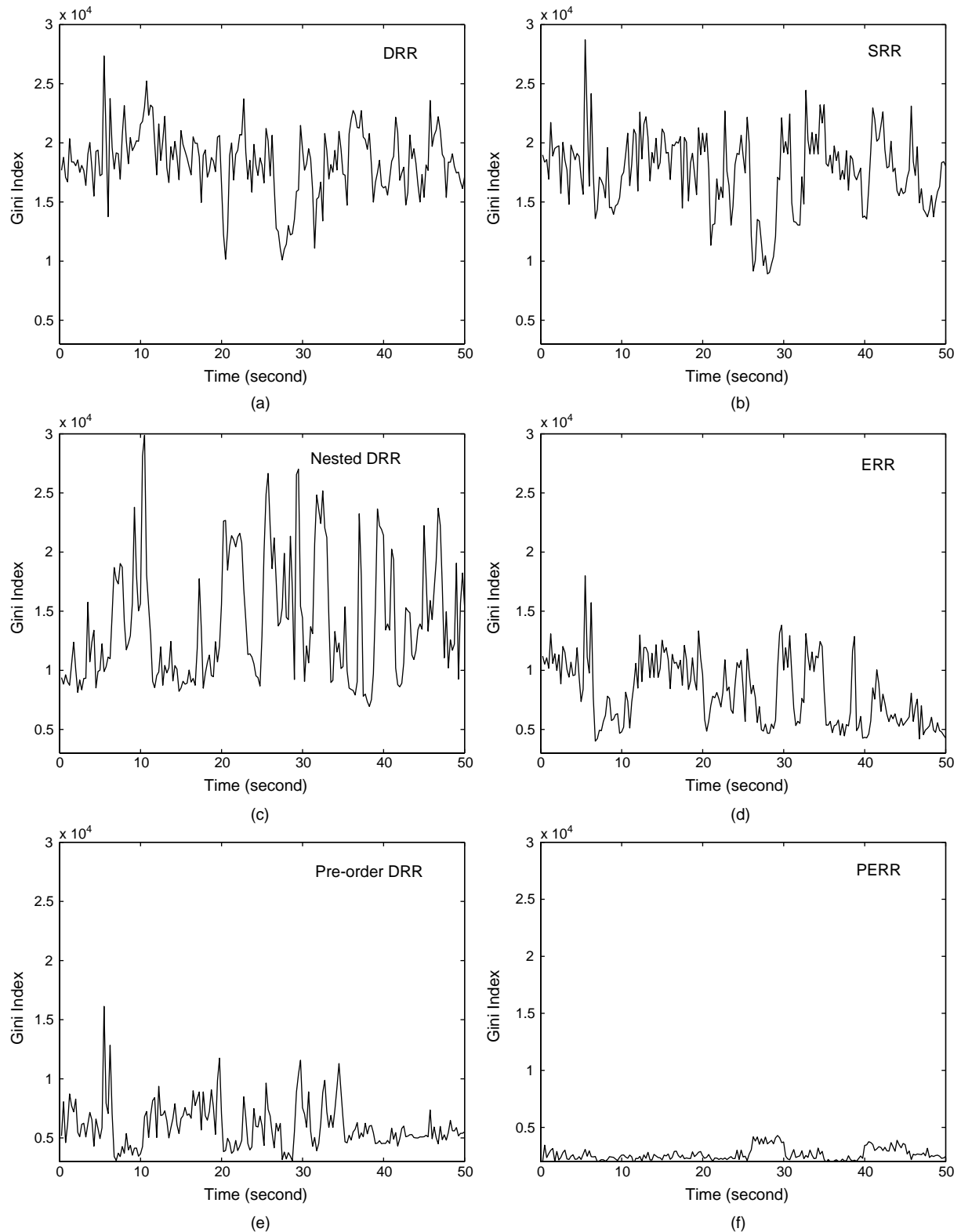


Fig. 7. Gini indices of frame-based packet schedulers with backbone router traffic traces.

a small value of the Gini index. With such a redefined optimization goal, the new scheduler is likely to achieve better end-to-end quality in the transmission of real-time multimedia traffic.

Acknowledgements

This work was supported in part by NSF CAREER Award CCR-9984161 and US Air Force Contract

F30602-00-2-0501. A preliminary version of this paper appeared in Proc. IEEE Int'l Performance, Computing, and Communications Conference (IPCCC), April 2003, Phoenix, AZ, USA.

References

- [1] D.C. Stephens, J.C.R. Bennett, H. Zhang, Implementing scheduling algorithms in high-speed networks, *IEEE J. Selected Areas Commun.* 17 (6) (1999) 1145–1158.
- [2] Cisco Systems, Inc., Cisco 12016 gigabit switch router, Application Note, 1999.
- [3] D. Bertsekas, R. Gallager, *Data Networks*, second ed., Prentice-Hall, Englewood Cliffs, NJ, 1992.
- [4] S. Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network*, Addison-Wesley, Reading, MA, 1997.
- [5] A.K. Parekh, R.G. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the single-node case, *IEEE/ACM Trans. Netw.* 1 (1) (1993) 344–357.
- [6] Y. Zhou, H. Sethu, On the relationship between absolute and relative fairness bounds, *IEEE Commun. Lett.* 6 (1) (2002) 37–39.
- [7] S.R. Golestani, A self-clocked fair queueing scheme for broadband applications, in: *IEEE INFOCOM*, 1994 pp. 636–646.
- [8] P. Goyal, H.M. Vin, H. Cheng, Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks, *IEEE/ACM Trans. Netw.* 5 (5) (1997) 690–704.
- [9] D. Stiliadis, A. Varma, Efficient fair queueing algorithms for packet-switched networks, *IEEE/ACM Trans. Netw.* 6 (2) (1998) 175–185.
- [10] M. Shreedhar, G. Varghese, Efficient fair queuing using deficit round-robin, *IEEE/ACM Trans. Netw.* 4 (3) (1996) 375–385.
- [11] S.S. Kanhere, H. Sethu, A.B. Parekh, Fair and efficient packet scheduling using elastic round robin, *IEEE Trans. Parallel Distrib. Syst.* 13 (3) (2002) 324–336.
- [12] J.A. Cobb, M.G. Gouda, A. El-Nahas, Time-shift scheduling—fair scheduling of flows in high-speed networks, *IEEE Trans. Netw.* 6 (3) (1998) 274–285.
- [13] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, in: *ACM SIGCOMM*, 1989 pp. 1–12.
- [14] J.C.R. Bennett, H. Zhang, WF²Q: worst-case fair weighted fair queueing, in: *IEEE INFOCOM*, 1996 pp. 120–128.
- [15] S. Floyd, V. Jacobson, Link-sharing and resource management models for packet networks, *IEEE/ACM Trans. Netw.* 3 (4) (1995) 365–386.
- [16] S. Floyd, Notes on class-based-queueing and guaranteed service, Unpublished notes, <http://www.aciri.org/floyd/cbq.html>, July 1995.
- [17] S.S. Kanhere, H. Sethu, Fair, efficient and low-latency packet scheduling using nested deficit round robin, in: *Proceedings of the IEEE Workshop on High Performance Switching and Routing*, 2001 pp. 6–10.
- [18] S. Tsao, Y. Lin, Pre-order deficit round robin: a new scheduling algorithm for packet-switched networks, *Comput. Netw.* 35 (2/3) (2001) 287–305.
- [19] S.S. Kanhere, Design and analysis of fair, efficient and low-latency schedulers for high-speed packet-switched networks, PhD thesis, Drexel University, Philadelphia, PA 19104, USA, <http://www.ece.drexel.edu/CCL/>, 2003.
- [20] A.K. Parekh, A generalized processor sharing approach to flow control in integrated services networks, PhD thesis, Massachusetts Institute of Technology, Feb. 1992.
- [21] D. Stiliadis, A. Varma, Latency-rate servers: a general model for analysis of traffic scheduling algorithms, *IEEE/ACM Trans. Netw.* 6 (5) (1998) 611–624.
- [22] F.A. Cowell, *Measuring Inequality: Techniques for the Social Sciences*, Wiley, New York, 1977.
- [23] A.W. Marshall, I. Olkin, *Inequalities: Theory of Majorization and its Applications*, Academic Press, New York, 1979.
- [24] A. Kumar, J. Kleinberg, Fairness measures for resource allocation, in: *41st Annual Symposium on Foundation of Computer Science*, IEEE, 2000. pp. 75–85.
- [25] J.E. Stiglitz, *Economics*, Norton, New York, 1993.
- [26] D. Stiliadis, A. Varma, Rate-proportional servers: a design methodology for fair queueing algorithms, *IEEE/ACM Trans. Netw.* 6 (2) (1998) 164–174.
- [27] National Laboratory for Applied Network Research, Passive Measurement and Analysis, <http://pma.nlanr.net/PMA/>.