Why WFQ Is Not Good Enough For Integrated Services Networks

Jon C. R. Bennett

FORE Systems Inc.
Warrendale, PA 15086

Hui Zhang

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Weighted Fair Queueing has received a lot of attention in the research community recently. In fact, the IETF has proposed to use WFQ as a basic building block for future integrated services networks. The reason for such a great interest is that it closely approximates a hypothetical fluid model discipline called Generalized Processor Sharing (GPS), which provides a conceptually ideal building block for integrated services network. There is a general misconception that WFQ provides almost identical service *to GPS with a maximum difference of one packet, thus it is an ideal and practical replacement for GPS. This is a misinterpretation and over-generalization of Parekh's result that the delay bound provided by WFQ is within one packet transmission time of that provided by GPS. In this paper, we will show that, contrary to popular belief, there could be large discrepancies between the services provided by the packet WFQ system and the fluid GPS system. The inaccuracy introduced by WFQ in approximating GPS are detrimental to both real-time and best-effort traffic in a network with hierarchical link-sharing service. In addition, WFQ has a relatively high complexity which make it difficult to implement. We propose two new algorithms and show that they overcome the limitations of WFQ. Simulation results are presented to demonstrate the advantages of the new algorithms.*

1 Introduction

Future integrated services networks [6, 8, 16] will support multiple services that include guaranteed real-time service, predicted real-time service, best-effort service, and others. In addition, it needs to support link-sharing [9], which allows resource sharing among applications that require different network services but belong to the same administrative class. Consider the example shown in Figure 1 (a). There are 11 agencies or organizations sharing the same output link. The administrative policy dictates that Agency A1 gets at least 50% of the link bandwidth when the network is overloaded. In addition, to avoid the starvation of best-effort traffic, from the 50% bandwidth assigned to A1, best-effort traffic should get at least 20% bandwidth. It is important to design mechanisms to meet the goals of link sharing and requirements of different service classes simultaneously.

Recently, Weighted Fair Queueing or WFQ has received much attention. WFQ is a packet approximation algorithm of the hypothetical Generalized Processor Sharing discipline. With GPS, there is a separate FIFO queue and a pre-specified service share for each session sharing the same link. During any time interval when there are exactly N non-empty queues, the server services the N packets at the head of the queues simultaneously, in proportion to the service shares of their corresponding sessions.

The link-sharing structure in Figure 1 (a) can be easily supported by a hierarchical GPS (H-GPS) as illustrated in Figure 1 (b). In addition, it has been shown that with a one-level GPS (1) an end-to-end delay bound can be provided to a session if the traffic on that session is leaky bucket constrained [12]; (2) bandwidth is fairly distributed to competing sessions [7] and sources can accurately estimate the available bandwidth to them in a distributed fashion [11]. The first property forms the basis for supporting the real-time traffic [6] and the second property enables robust and distributed end-to-end traffic management algorithms for best-effort traffic [11, 14]. Having a hierarchical GPS affects only the distribution of excess bandwidth unused by each subclass, but not the other two properties. Therefore, the simple H-GPS configuration in Figure 1 (b) supports the three goals, namely, link-sharing, real-time traffic management, and best-effort traffic management.

The above example shows that GPS is an ideal building block for integrated services network. Since GPS is a hypothetical fluid system that can not be implemented in real world, packet algorithms that approximate GPS have been proposed.

Among them, WFQ is the most well-known algorithm. Parekh's seminal work [12] shows that in the absence of link-sharing, the end-to-end delay bound provided by Weighted Fair Queueing (WFQ) [7, 12], which is the standard packet approximation algorithm of GPS, is very close to that provided by GPS. While WFQ maintains the bounded delay property of GPS, its fairness property is much weaker than GPS. As we will demonstrate in this paper, WFQ can introduce substantial inaccuracy in approximating GPS. We will show that such inaccuracy will significantly affect best-effort traffic management, real-time traffic management, and link-sharing algorithms.

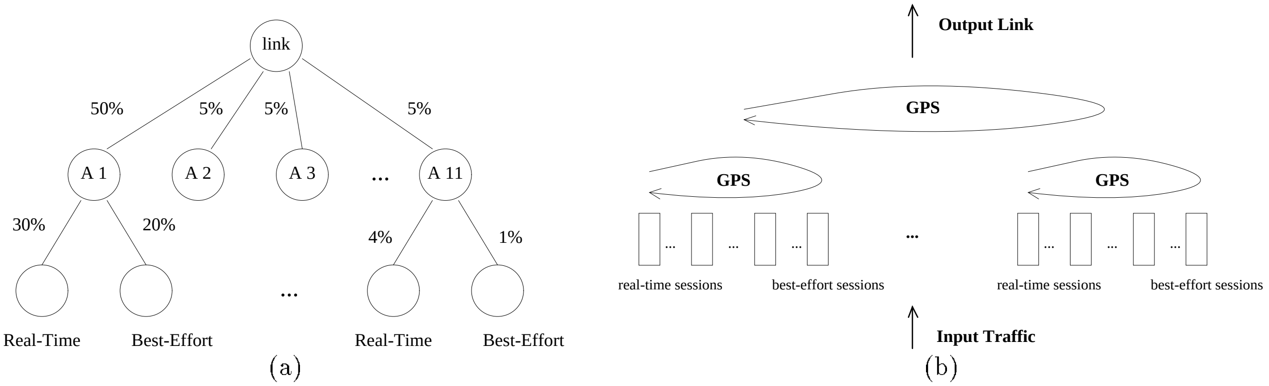


Figure 1: A Link Sharing Example

2 Limitations of WFQ

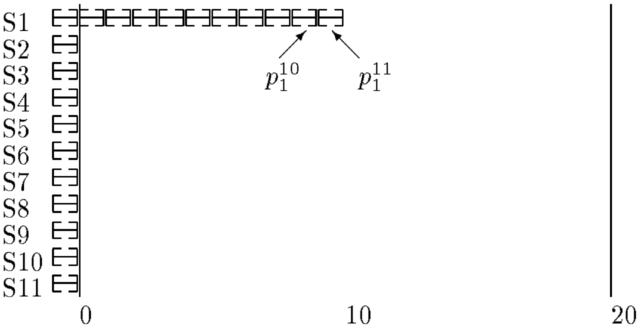


Figure 2: Packet Arrivals

Weighted Fair Queueing (WFQ) [6, 7], also known as the Packet-by-Packet Generalized Processor Sharing [12], is the most well-known packet approximation algorithm for GPS. In WFQ, when the server is ready to transmit the next packet at time τ, it picks, among all the packets queued in the system at τ, the first packet that would complete service in the corresponding GPS system if no additional packets were to arrive after τ.

Parekh established the following important result on the services provided by WFQ and GPS: the delay bound provided by WFQ is within one packet transmission time of that provided by GPS [12]. This result forms the basis for providing real-time service using WFQ. In fact, within Internet Engineering Task Force, the standard body for Internet, WFQ has just recently been proposed as the reference server model for the guaranteed service class in the Internet [13]. However, this result can easily be mis-interpreted to state that the packet WFQ discipline and the fluid GPS discipline provide almost identical service except for a difference of one packet.

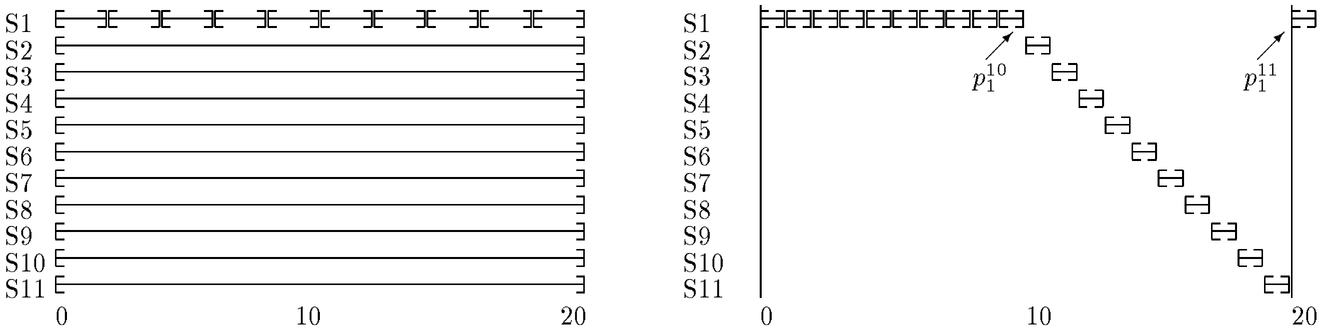
Contrary to this popular (but incorrect) belief, we will demonstrate below that there could be *large* discrepancies between the services provided by WFQ and GPS. In addition, we will show that such a large discrepancy significantly affects WFQ's fairness property, which are essential for the link-sharing algorithm and the end-to-end traffic management algorithm for best-effort service.

2.1 Inaccuracy of WFQ

Consider the example illustrated in Figure 2 where there are 11 sessions sharing the same link. The horizontal axis shows the time line and the vertical axis shows the sample path of each session. For simplicity, assume all packets have the same size of 1 and the link speed is 1. Also, let the guaranteed rate for session 1 be 0.5, and the guaranteed rate for each of the other 10 sessions be 0.05. In the example, session 1 sends 11 back-to-back packets starting at time 0 while each of all the other 10 sessions sends only one packet at time 0. If the server is GPS, it will take 2 time units to service each of the first 10 packets on session 1, one time unit to service the 11^{th} packet, and 20 time units to service the first packet from another session. Denote the k^{th} packet on session j to be p_j^k, then in the GPS system, the starting time is $2(k-1)$ for $p_1^k, k = 1 \ldots 10$, 21 for p_1^{11}, and 20 for $p_j^1, j = 2 \cdots 11$. This is shown in Figure 3 (a). Under WFQ, since the first 10 packets on session 1 ($p_1^k, k = 1 \ldots 10$) all have GPS finish times smaller than packets on other sessions, the server will service 10 packets on session 1 back to back before service packets from other sessions. After the burst, the next packet on session 1, p_1^{11}, will have a larger finishing time in the GPS system than the 10 packets at the head of other sessions' queues, therefore, it will not be serviced until all the other 10 packets are transmitted, at which time, another 10 packets from session 1 will be serviced back to back. This cycle of bursting 10 packets and going silent for 10 packet times can continue indefinitely.

As will be discussed below, this oscillation of service rate will significantly impact both (a) delay bounds for real-time traffic when there is hierarchical link sharing; and (b) traffic management algorithms for best-effort traffic.

The reason for this poor behavior of WFQ is that the amount of service WFQ provides to a session can



(a) GPS Service Order

(b) WFQ Service Order

Figure 3: Inaccuracy of WFQ in Approximating GPS

be much larger than that provided by GPS during certain period of time. In the above example, between time 0 and 10, WFQ serves 10 packets from session 1 while GPS serves only 5. After such a period, WFQ needs to serve other sessions in order for them to catch up. Intuitively, the difference between the amounts of service provided to each session by WFQ and the GPS is a measure of inaccuracy of WFQ in approximating GPS. Therefore, the inaccuracy introduced by WFQ is *not* merely one packet, but $N/2$ packets, where N is the number of sessions sharing the link.

2.2 Impact of Inaccuracy on Link-Sharing and Real-Time Service

While the delay bounds provided by WFQ and GPS are very similar without link-sharing, they can differ significantly with link sharing because of the large inaccuracy introduced by WFQ. Thus, the real-time service will be negatively affected by WFQ due to the much larger delay bounds.

Consider the example with a link sharing structure in Figure 1 (a) and the packet arrival sequence in Figure 2. Assume that WFQ discipline is used instead of GPS and the first 10 packets of class A1 belong to the best-effort sub-class and the 11^{th} packet belong to the real-time sub-class. Even though the real-time sub-class of A1 reserves 30% of the link bandwidth, when a real-time packet arrives, it may still have to wait at least 10 packet transmission times. Now consider the example where there are 1001 classes sharing a 100 Mbps link with the maximum packet size of 1500 bytes. For a real-time session reserving 30% of the link bandwidth, its packet may be delayed by 120 ms in just one hop!

On the other hand, if GPS is used, the end-to-end delay bound provided to each real-time session will be the same regardless whether there is link-sharing. Having a hierarchical GPS affects only the distribution of excess bandwidth unused by each subclass, but not the worst-case performance bounds for each session. In contrast, having a hierarchical WFQ will significantly affect the worst-case performance guarantees for each session, and the delay bound provided to a session by a hierarchical WFQ server can be much larger than that provided by a hierarchical GPS server.

The reason that hierarchical WFQ introduces a large delay bound for real-time traffic is that some packets *related to the real-time session* have received *more* service than deserved in the previous time period. In the case of a non-hierarchical server, these packets must belong to the same session; in the case of a hierarchical server, these packets may belong to sessions that share a subtree with the session. In the example above, when the real-time sub-class of A1 traffic is not present in the system, the packets from the best-effort sub-class of A1 not only can consume all the bandwidth reserved for class A1 in the current time period, as dictated by the hierarchical link-sharing policy, but also can get ahead and use bandwidth that are reserved for the future traffic. When a real-time sub-class packet from class A1 shows up, the total available bandwidth available to class A1 is in deficit, therefore, it has to wait for a long time period during which traffic from other classes can catch up with their fair service shares. Such a problem does not exist for GPS as no session can get more bandwidth than its fair share at any given time.

2.3 Impact of Inaccuracy on Feedback-based Algorithms

While accurate estimation of bandwidth available for each source in a dynamic network environment is a pre-requisite for an efficient and robust feedback-based traffic management algorithm, the oscillation of service rates introduced by the inaccuracy of WFQ will significantly affect the rate estimation algorithm and result in instability of end-to-end control algorithms. This is true for both rate-based and window-based flow control algorithms.

In [11], Keshav proposes a rate-based algorithm called Packet-Pair for estimating the available bandwidth for a source. In the packet-pair algorithm, the source sends two back-to-back probe packets and the receiver sends acknowledgement packets immediately upon receiving each packet. The source then uses the spacing between the two acknowledgement packets to calculate an estimate of the bottleneck server rate available to the session. In the example above, if the WFQ algorithm is used, the estimated available

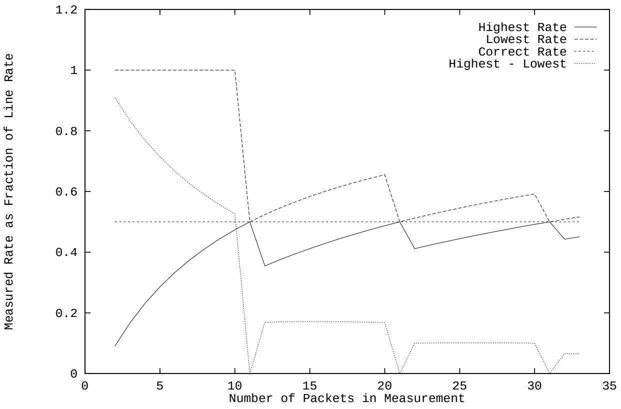


Figure 4: Upper and Lower bounds of measured rate for session 1

rate to session 1 will *oscillate* between full link speed and zero link speed. This is likely to cause instability of the source control algorithm.

To address the problem of measurement errors by the Packet-Pair algorithm, Bernstein proposes an enhancement of inserting data packets between the packet-pairs [3]. Even with such an enhancement, the measurement error still persists as shown in Figure 4. Depending whether the two probe packets are sent during the burst period or the silent period, the estimate of the server rate may range from r to $\frac{r}{N+1}$ where N is the number of other sessions. In particular, the bound on the measurement error is *not* a strictly decreasing function of the number of packets in the measurement interval.

For window-based flow control algorithms such as TCP, the stability of the algorithm depends on the accurate measurement and robust estimation of Round Rrip Times (RTT's). RTT measured by the source will be modulated by the available rate to the session, therefore will also oscillate significantly in this example.

2.4 Quantifying Inaccuracy Using Worst-case Fair Index

We have shown that WFQ can introduce significant inaccuracy in approximating GPS, and this inaccuracy can be detrimental to both real-time and non-real-time traffic in a hierarchical link-sharing environment. In this section, we propose to use a metric called Worst-case Fair Index (WFI) to characterize the fairness property of Packet Fair Queueing (PFQ) servers.

Definition 1 *A server s is said to guarantee a Time Worst-case Fair Index (T-WFI) off $\mathcal{A}_{i,s}$ for session i, if for any time τ, the delay of a packet arriving at τ is bounded above by $\frac{1}{r_i}Q_i(\tau) + \mathcal{A}_{i,s}$, that is,*

$$d_i^k - a_i^k \leq \frac{Q_i(a_i^k)}{r_i} + \mathcal{A}_{i,s} \qquad (1)$$

where r_i is the throughput guarantee to session i $Q_i(\tau)$ is the number of bits in the session queue at time τ (including the packet that arrives at time τ), a_i^k and d_i^k are the arrival and departure times of k^{th} packet of session i respectively.

We call $\mathcal{A}_{i,s}$ the Time Worst-case Fair Index (T-WFI) for session i at server s at it characterizes the worst-case fair property in unit of time. As discussed in [1] WFI can also be represented in unit of bits. Intuitively, $\mathcal{A}_{i,s}$ represents the maximum time a packet coming to an empty session queue needs to wait before start receiving its guaranteed service rate. An important observation is that both GPS and H-GPS have a WFI of 0. That is, with GPS or H-GPS, a packet coming an empty session queue can receive its guaranteed service rate immediately after its arrival. However, as we see in the example shown in Section 2, the T-WFI for WFQ can be quite large. In fact, it has been shown in [2] that the WFI for WFQ can increase linearly as a function of the number of sessions sharing the link.

In [1], we have shown that the delay bound that can be guaranteed by a Hierarchical Packet Fair Queueing server to session i is a increasing function of T-WFI of the Packet Fair Queueing servers in the hierarchy. To achieve a tight delay bound in a hierarchical server, the root or intermediate Packet Fair Queueing server in the link-sharing hierarchy needs to have small WFI's. WFQ has a large WFI, therefore, the delay bound provided by H-WFQ is rather large.

2.5 Implementation Complexity of WFQ

An implementation of WFQ based on a virtual time function $V_{GPS}(\cdot)$ is proposed in [7, 12]. When a packet arrives, it is stamped with a virtual finish time. Packets are transmitted in the increasing order of virtual finish times. There are two types of cost associated with such an implementation: sorting packets and computing $V_{GPS}(\cdot)$. To pick the packet with the smallest virtual finish time, a sorted priority based on packet's virtual finish time needs to be maintained. This has a complexity of O(log(N)), where N is the number of sessions sharing the link. While a number of implementation techniques have been proposed to speed up the sorting [4, 5], it is unclear how to compute $V_{GPS}(\cdot)$ efficiently. The crux of the problem is that the virtual service function needs to keep track of the process of the GPS. In a fluid system, a server can serve packets from all backlogged sessions simultaneously, it is possible that all N packets finish service within an arbitrary short period of time, which means that in the worst case the scheduler needs to process N events for a single scheduling decision. This makes WFQ difficult to implement at high speed.

3 WF^{2}Q and WF^{2}Q+

In Section 2, we have shown that WFQ has two important limitations that make it inadequate to be the basic building block for integrated services networks: first, it introduces significant inaccuracies in approximating GPS. Second, it has a high implementation complexity due to the need of emulating GPS.

In [1, 2], we propose two new packet approximation algorithms of GPS that overcome the limitations of WFQ. The Worst-case Fair Weighted Fair Queueing (WF^{2}Q) is an optimal packet algorithm in terms of accuracy in approximating the fluid GPS — the service provided by WF^{2}Q is almost identical to that of GPS, differing by no more than one maximum size packet. The WF^{2}Q+ algorithm maintains all the important properties of WF^{2}Q, but has a lower degree of complexity.

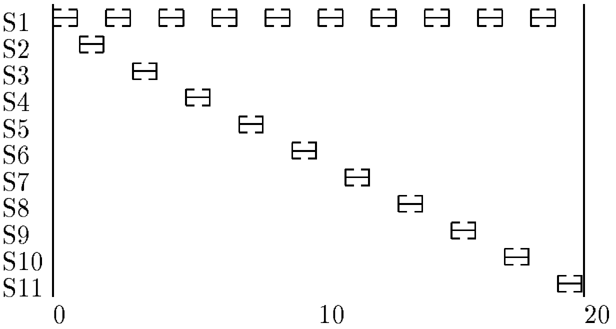


Figure 5: WF^{2}Q Service Order

In a WF^{2}Q system when the next packet is chosen for service at time τ, rather than selecting it from among all the packets at the server as in WFQ, the server only considers the set of packets that *have started* (and possibly finished) receiving service in the corresponding GPS system at time τ, and selects the packet among them that would complete service first in the corresponding GPS system. If we consider again the example shown in Figure 2,

Now consider again the example discussed in Figure 2 but with WF^{2}Q policy. at time 0, all packets at the head of each session's queue, p_i^1, $i = 1, \cdots, 11$, have started service in the GPS system (Figure 3 (a)). Among them, p_1^1 has the smallest finish time in GPS, so it will be served first in WF^{2}Q. At time 1, there are still 11 packets at the head of the queues: p_1^2 and p_i^1, $i = 2, \cdots, 11$. Although p_1^2 has the smallest finish time, it will not start service in the GPS system until time 2, therefore, it won't be eligible for transmission at time 1. The other 10 packets have all started service at time 0 at the GPS system, thus are eligible. Since they all finish at the same time in the GPS system, the tie-breaking rule of giving highest priority to the session with the smallest number will yield p_2^1 as the next packet for service. In contrast, if a WFQ server is used, rather than selecting the next packet from among the 10 packets that have started service in the GPS system, it would pick the packet among all 11 packets, which will result in packet p_1^2 . At time 3, p_1^2 becomes eligible and has the smallest finish time among all backlogged packets, thus it will start service next. The rest of the sample path for the WF^{2}Q system is shown in Figure 5. The inaccuracy introduced by WF^{2}Q in this example is one packet.

In general, the following holds.

- the difference of services provided by WF^{2}Q and GPS is bounded by one maximally sized packet;
- for a leaky bucket constraint source, a WF^{2}Q system provides the same end-to-end delay bound as a corresponding WFQ system

Since any packet system will introduce an inaccuracy of at least one packet, this means that WF^{2}Q is the *optimal* packet algorithm in terms of accuracy in approximating GPS. In addition, the normalized T-WFI for WF^{2}Q is the service time of one maximum size packet. Since the normalized worst-case fair index for a packet system is at least one packet transmission time, this means that WF^{2}Q is also an optimal packet policy with respect to the worst-case fair property.

While WF^{2}Q is the most accurate packet algorithm to approximate GPS, it still has the same implementation complexity as WFQ. The key problem is that both WF^{2}Q and WFQ need to compute $V_{GPS}(\cdot)$, which is equivalent to emulating the GPS system. In [1], we introduce a refined algorithm called WF^{2}Q+ . WF^{2}Q+ differs from WF^{2}Q in that it computes a virtual time function $V_{WF^2Q+}(\cdot)$ without emulating GPS.

With WF^{2}Q+ , the virtual time function is defined as follows,

$$V_{WF^2Q+}(t+\tau) = \max(V_{WF^2Q+}(t)+\tau, \min_{i \in B_{WF^2Q+}(t)} (S_i^{h_i(t)}) \qquad (2)$$

where $B_{WF^2Q+}(t)$ is the set of sessions backlogged in the WF^{2}Q+ system at time t, $h_i(t)$ is the sequence number of the packet at the head of the session i's queue, and $S_i^{h_i(t)}$ is the virtual start time of the packet. Notice that the computation of the virtual time function does not need the emulation of GPS.

A number of algorithms have been proposed to approximate $V_{GPS}(\cdot)$ without explicitly emulating GPS, for example, the Self-Clocked Fair Queueing (SCFQ) [10] and the Framed Based Fair Queuing [15]. However, none of them achieves enough accuracy and the resulted Packet Fair Queueing algorithms have large WFI's.

WF^{2}Q has the following set of salient properties of WF^{2}Q+ . It

- provides the tightest delay bound among all PFQ algorithms (so does WFQ and WF^{2}Q, but not SCFQ);
- has the smallest WFI among all PFQ algorithms (so does WF^{2}Q, but not WFQ or SCFQ);
- having relative low implementation complexity of O(log(N) (so does SCFQ, but not WFQ or WF^{2}Q);

- needs only per session instead of per packet timestamp (so does SCFQ, but not WFQ or WF^2Q);
- and is work-conserving.

We refer the reader to [1] for the details of the proof.

4 Simulation Experiments

In this section, we present results based on simulation experiments to illustrate the impact of WFQ's inaccuracy on traffic management algorithms. For our simulation experiments the service hierarchy used is shown in Fig 6. Each node in the figure is labeled as follows, the rate above the node is the guaranteed service rate of the node. The value inside the node represents the node's guaranteed rate as a fraction of it's parent's rate. If the node is a leaf node, the label below identifies the session at that leaf, if the node is an interior node, it has a reference label at its side. For simplicity, all packets are 8 KB.

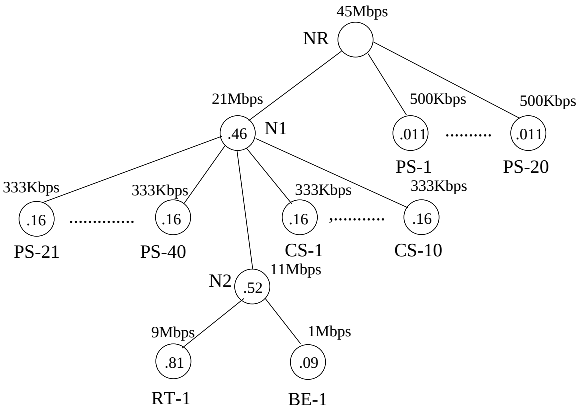


Figure 6: Hierarchical Service Allocation

In the simulations we will be interested in the service received by sessions RT-1 and BE-1. Real-time session RT-1 is a deterministic on-off source. It transmits at 9Mbps with a 50 ms on period and 50 ms off period. For session RT-1 we will be interested in the delay and delay variation experienced. Session BE-1 is a best effort session, running TCP[1]. Session BE-1 should be able to use the bandwidth guaranteed to session RT-1, when RT-1 is idle. For session BE-1, we will be observing the interpacket spacing , to see if the feedback received by the best effort session can be used to determine the state of the network. The sessions labeled PS-N are poisson sources transmitting at 150% of their guaranteed bandwidth. They act as a set of persistently backloged connections, whose start times are randomized. Sessions CS-N are correlated deterministics sources transmitting at 150% of their guaranteed rate. These sessions are used to explore the effects that can occur when even a small percent of the total traffic is correlated.

Figure 7 (a) shows the packet delay experienced by session RT-1 under a hierarchical WFQ server with the service allocation illustrated in figure 6. Figure 7 (a) clearly shows that not only is the largest delay experienced (24ms) much larger than the worst case delay under GPS of $\frac{8\mathrm{KB}}{9\mathrm{Mbps}} =\sim 7ms$, but that the delay jitter is nearly as large.

Figure 7 (b) shows the inter-packet spacing of the TCP acks for session BE-1 that shares node N2 with the real-time session RT-1. What we would expect to see are spikes in the packet spacing every 100ms (on the 100ms marks) which last for 50ms as the bandwidth available to session BE-1 is reduced during the period when session RT-1 is active. During the periods between the spikes the inter-packet spacing should be relatively constant, as there are no other fluctuations in the bandwidth available to session BE-1. Instead we see spikes that are much larger than expected, and we see that the spikes do not correspond well with the session RT-1 active periods. We also see a number of instances of very small inter-packet spacing, indicating the reception of back-to-back packets. This will result in first the underestimation, then the overestimation of bandwidth by a Packet-Pair like measurement technique.

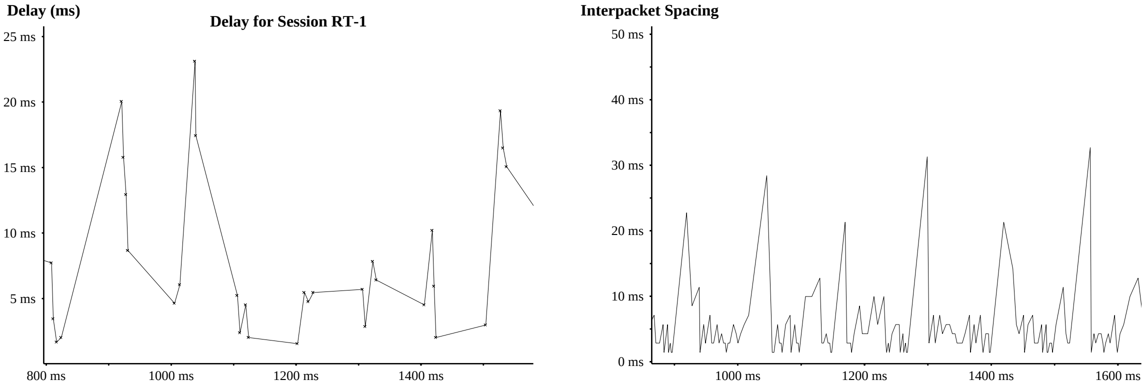
Figure 8 shows the same experiment but using WF^2Q+ instead. Comparing Figure 7(a) and Figure 8(b), we can see that the worst-case delay under $H\text{-}WF^2Q+$ is much smaller than that under $H\text{-}WF^2Q$.

In Figure 8 (b) we see almost exactly the behavior that we would expect, indicating that session BE-1 is receiving almost precisely the correct bandwidth at all times. It is clear that the bandwidth available to BE-1 could almost be estimated without doing any averaging at all. Making $H\text{-}WF^2Q+$ ideal for use with both, data transport protocols which attempt to estimate the available bandwidth, and real time traffic at the same time.

In Figure 9, we repeat the same experiment again, but with the other popular packet approximation of GPS called Self-Clocked Fair Queueing, also known as "Chuck's Approximation". We see in Figure 9 (a) that the worst case delay seen by RT-1 is slightly larger then when we used WFQ. More strikingly, the spikes in Figure 9 (b) are much larger then in Figure 7 (b), and there many more occurrences of back-to-back packets. So we see that the oscillations experienced by both real-time and best-effort traffic under H-SCFQ are slightly worse than under H-WFQ, and much worse than under $H\text{-}WF^2Q+$. Showing that $H\text{-}WF^2Q+$ is far better suited for use in integrated services networks.

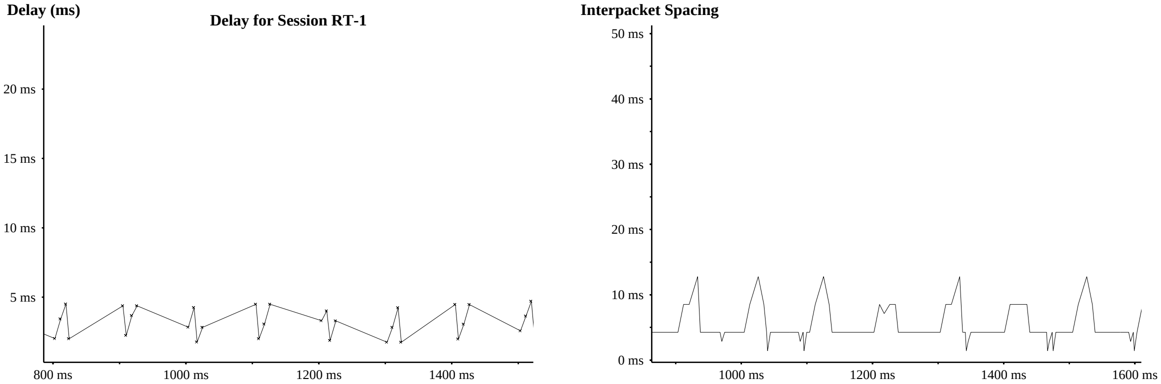
---

[1] The TCP used in the simulation immediately acks all packets, the same as Packet-Pair



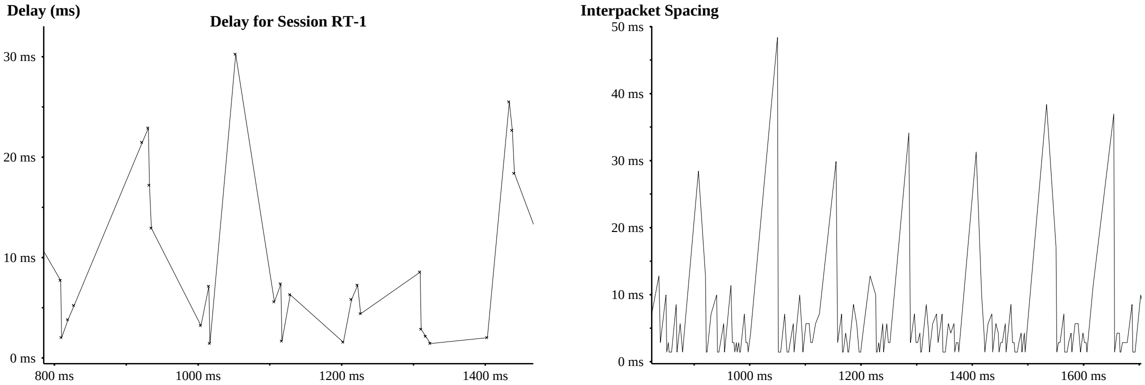
(a) Delay of Real-Time Session RT1 (b) Packet Spacing for Session BE-1

Figure 7: Hierarchical WFQ



(a) Delay of Real-Time Session RT1 (b) Packet Spacing of Session BE-1

Figure 8: Hierarchical WF^{2}Q+



(a) Delay of Real-Time Session RT1 (b) Packet Spacing of Session BE-1

Figure 9: Hierarchical SCFQ

5 Summary

While WFQ has been widely considered to be the most accurate packet approximation algorithm for GPS, we showed in this paper that there could be large discrepancies between the services provided by WFQ and GPS. The inaccuracy introduced by WFQ can (a) significantly increase the delay bound for real-time sessions under hierarchical link-sharing; (b) and cause end-to-end feedback algorithms for best-effort traffic to oscillate. We proposed to use a metric called Worst-case Fair Index to characterize the accuracy of a Packet Fair Queueing algorithm in approximating GPS. We presented two algorithms, WF^{2}Q and WF^{2}Q+ , which have the smallest WFI among all Packet Fair Queueing algorithms. WF^{2}Q+ has the added advantage in that it has a lower complexity than WFQ and WF^{2}Q.

References

[1] J.C.R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. to appear in ACM SIGCOMM'96.

[2] J.C.R. Bennett and H. Zhang. WF^{2}Q: Worst-case fair weighted fair queueing. In *IEEE INFOCOM'96*, San Francisco, March 1996.

[3] Greg M. Bernstein. Reserved bandwidth and reservationless traffic in rate allocating servers. *Computer Communication Review*, 23(3), July 1993.

[4] R. Brown. Calendar queues: A fast O(1) priority queue implementation for the simulation event set problem. *Communications of the ACM*, 31(10):1220–1227, October 1988.

[5] H. Chao. Architecture design for regulating and scheduling user's traffic in ATM networks. In *Proceedings of ACM SIGCOMM'92*, pages 77–87, Baltimore, Maryland, August 1992.

[6] D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceedings of ACM SIGCOMM'92*, pages 14–26, Baltimore, Maryland, August 1992.

[7] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Journal of Internetworking Research and Experience*, pages 3–26, October 1990. Also in Proceedings of ACM SIGCOMM'89, pp 3-12.

[8] D. Ferrari and D. Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications*, 8(3):368–379, April 1990.

[9] S. Floyd and V. Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM Transactions on Networking*, 3(4), August 1995.

[10] S. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proceedings of IEEE INFOCOM'94*, pages 636–646, Toronto, CA, June 1994.

[11] S. Keshav. A control-theoretic approach to flow control. In *Proceedings of ACM SIGCOMM'91*, pages 3–15, Zurich, Switzerland, September 1991.

[12] A. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks.* PhD dissertation, Massachusetts Institute of Technology, February 1992.

[13] S. Shenker R. Braden, D. Clark. Integrated services in the internet architecture: an overview, June 1994. Internet RFC 1633.

[14] S. Shenker. Making greed work in networks: A game theoretical analysis of switch service disciplines. In *Proceedings of ACM SIGCOMM'94*, pages 47–57, London, UK, August 1994.

[15] D. Stilliadis and A. Verma. Frame-based fair queueing: A new traffic scheduling algorithm for packet-switched networks. Technical Report USCS-CRL-95-39, University of California at Santa Cruz, July 1995.

[16] J. Turner. New directions in communications(or which way to the information age?). *IEEE Communication Magazine*, 24(10), October 1986.