Quality of Service Driven Packet Scheduling Disciplines for Real-time Applications: Looking Beyond Fairness

David A. Hayes and Michael Rumsewicz
Software Engineering Research Centre,
The Royal Melbourne Institute of Technology,
Carlton Victoria 3053 AUSTRALIA
{david, mpr}@serc.rmit.edu.au

Lachlan L. H. Andrew
Department of Electrical and Electronic Engineering,
University of Melbourne,
Parkville Victoria 3052 AUSTRALIA
l.andrew@ee.mu.oz.au

Abstract—**In this paper we focus on real-time scheduling of "soft" real-time data services such as multimedia data, MPEG video streaming and IP telephony, which can tolerate a small degree of loss or delay. We argue that network operators and service providers should be able to select from a range of Quality of Service objectives, including maximizing the number of customers receiving good service. Further, we argue that scheduling disciplines such as fair queueing are unable to achieve such goals and hence there is a need for alternative approaches. We propose a new scheduling scheme, which we call the Dual Queue discipline. We show that the Dual Queue has the flexibility to satisfy a variety of QoS objectives, ranging from existing notions of fairness through to maximizing the number of customers receiving good service. In addition, even the simplest Dual Queue implementation outperforms Fair Queueing, is scalable in the number of active sessions, and can be made fair, if desired, over moderate to long time scales.**

Keywords—**Quality of Service, packet scheduling, Dual Queue, real-time packet scheduling.**

I. Introduction

An increasing amount of traffic on packet switched networks is time-critical. This traffic includes traditional real-time services such as telephony as well as new services such as video on demand, multicast video and virtual reality applications [1], [2], [3]. For such applications to be successful, networks must provide sufficient Quality of Service (QoS) to user sessions with respect to several quality measures, such as packet delay and loss rates. Guarantees with respect to QoS can be achieved through a combination of effective admission control and real-time scheduling.

However, the services mentioned above typically have relatively bursty traffic profiles. Further, at session setup customers are not usually able to accurately describe their traffic profile. Therefore, unless peak bandwidth admission control is employed, it is inevitable that moments of transient congestion will occur in the network.

In this paper we focus on congestion management of "soft" real-time services [4], such as multimedia data, MPEG video streaming and IP telephony, which can tolerate a small degree of loss or delay, rather than those requiring guaranteed delivery and delay bounds [5], [6], such as mission critical applications.

Our aims in this paper are two-fold:

- To highlight shortcomings in existing approaches to real-time scheduling with respect to QoS, and
- To provide a scheduling infrastructure which allows service providers and network operators to select and implement their own QoS objectives.

The novel aspect of our work is that we are able to support a range of QoS goals, from traditional concepts of fairness through to maximizing the number of users receiving excellent service over their entire session. To our knowledge, the latter is not addressed by existing scheduling disciplines.

When a system is lightly loaded, the QoS requirements of all sessions can usually be met. During congestion episodes, some or all sessions will receive service which does not meet their QoS requirements. Hence, it is necessary to decide which traffic streams should receive degraded service, either in terms of loss, delay or both. This is the problem of real-time bandwidth scheduling.

A common thread in the literature is to assign bandwidth fairly (see, for example, [2], [3], [7]) in which users with moderate bandwidth requirements are not penalized because of the excessive demands of others. This is the concept of fair queueing [10], [2], [3], [8], [9], in which the available bandwidth is shared equally among all competing sessions. As a result of equalizing the bandwidth, these schemes typically provide satisfactory QoS for sessions whose bandwidth requirements are less than their fair share. Fair queueing has since been enhanced to allow for weighted assignment of bandwidth, thereby allowing more general access to available bandwidth by traffic streams.

Fairness, however, is not a direct measure of Quality of Service as perceived by the user. Fairness is a measure of how the network assigns resources during periods of congestion, a measure the user is not able to perceive directly; it is not a guarantee of good service quality.

We contend that more relevant measures of QoS are the proportion of customers receiving poor service at any time during their session or during a congestion episode because this reflects the number of customers that might complain to their service provider. It follows, therefore, that a scheduling discipline should strive to maximize the number of customers receiving good service, rather than strive for a fairness of which the customers are unaware. Such a scheduling discipline should, however, provide the mechanisms to enforce fairness as well if the service provider deems it important. In this way, decisions on the desirability of fairness in a network may be left to the service providers as a business decision.

In this paper, then, we present a new approach to the management of transient congestion; rather than aiming to maximize fairness and letting good performance follow, we aim to provide an infrastructure that allows selectable QoS measures, including maximizing the number of users receiving good service. We achieve this aim by selecting certain sessions to bear the brunt of degraded service during periods of congestion. On short time scales, the approach may be very unfair, giving very poor service to a small number of sessions in order to provide sufficient resources for the remaining sessions. However, we construct the scheme such that fairness can be attained over moderate time scales, of the order of seconds, and still provide substantially

0-7803-5417-6/99/$10.00 ©1999 IEEE.

better overall quality of service than Fair Queueing (FQ) disciplines.

We call our scheduling discipline the Dual Queue scheduling discipline. One queue is used for the imminent transmission of packets, while the other is use to store packets of sessions to receive degraded service.

We show that

- Fair Queueing scheduling disciplines are unable to achieve QoS objectives such as maximizing the proportion of users receiving good service at any point in time or over the the lifetime of user sessions. Such objectives may be of significant interest to network operators.
- The Dual Queue scheduling discipline presented here has the flexibility to satisfy a variety of QoS objectives, ranging from existing notions of fairness through to maximizing the number of customers receiving good service.
- Even the simplest Dual Queue implementation can outperform Fair Queueing in providing good quality of service to users of real-time applications.
- The Dual Queue approach is scalable in the number of active sessions.
- The Dual Queue approach while unfair on small time scales can be made fair, if desired, over moderate to long time scales.

The rest of this paper is organized as follows. Some of the existing approaches to bandwidth allocation are discussed in Section II, along with a discussion of the approach we are proposing. A more detailed description of the Dual Queue Scheduling Discipline follows in Section III. Section IV describes the simulation scenario used to determine the performance of the proposed algorithm, and simulation results are presented in Section V. The implications of these results will be discussed in Section VI, and conclusions presented in Section VII.

II. Approaches to Quality of Service Scheduling for Real-time Packet Applications

As discussed in [8], traffic scheduling should possess certain desirable characteristics, including: isolation of flows, low end-to-end delays, efficient use of available bandwidth, Simplicity of implementation, scalability, and fairness. Many scheduling disciplines have been designed in an attempt to meet these requirements, but the prevailing wisdom is towards the use of Fair Queueing.

The concept of Fair Queueing (FQ) was introduced by Demers, Keshav and Shenker [10]. Fair queueing attempts to give each session its fair share of the available bandwidth. Fair queueing in essence tries to make the link look to each session like a processor sharing server, giving each session equal access to the available bandwidth. The concept generalizes to Weighted Fair Queueing schemes in which the scheduler may assign different weights to each session. This approach has spawned a vast range of adaptations, each seeking to improve on minor perceived shortcomings in fairness, implementation detail or scalability in the original Weighted Fair Queueing approach.

However, in all of these variants, fairness is an intrinsic goal of the discipline, and the idea of fairness is applied on very small time scales.

The focus on fairness at small time scales tends to obscure the fact that fairness is an attempt at inferring user QoS from quantities that may be directly measured by the network, rather than an actual measure of the user perceived QoS.

Fairness is *not* a direct measure of user perceived QoS because a user is unable to determine the Quality of Service received by other users, nor the bandwidth received by other users. Most users are unlikely to care about the service received by others, being motivated more directly by maximizing their own QoS.

Consider, now, the position of a network operator or service provider. After profit, it is most motivated by minimizing the number of complaints they receive from their customers concerning its perceived QoS. Thus, the network and the scheduling disciplines it employs should attempt to maximize the number of users receiving acceptable service.

Scheduling disciplines such as FIFO and FQ do not achieve this aim. For FIFO systems, a single misbehaving session can result in all traffic streams receiving poor service (measured by fraction of packets discarded or excessively delayed per traffic stream). For FQ systems, if all traffic streams burst simultaneously, all are treated equally and all receive degraded service.

Thus, we contend that fairness on very small time scales is *not* necessarily a good measure of the performance of a scheduling discipline for real-time services.

We conclude that network operators should be able to choose their own criteria for measuring network performance, including maximizing the number of satisfied customers. Thus, the network scheduling disciplines used should have the flexibility to allow the network operator to put into practice its own business philosophy.

In the following section, we describe a new scheduling discipline for real-time services with this property. The essence of our approach is to provide a scheduling discipline infrastructure which allows network operators to choose whether to provide fair service to all customers or to maximize the number of customers receiving acceptable service, or somewhere in between. We achieve this by recognizing when service degradation is about to occur and then deliberately selecting some traffic sessions to bear the brunt of degraded service until the transient congestion episode has passed. The specific selection process employed governs whether fairness or maximizing the number of satisfied customers is the major goal of the network operator.

III. The Dual Queue Scheduling Discipline

The Dual Queue approach, depicted in Figure 1, uses two queues to implement the philosophy described in Section II. We refer to these as the α- and β-queues.

The α-queue is a short First In-First Out (FIFO) queue. Its length corresponds to the longest delay that can be considered "good service" in terms of delay. The β-queue may be significantly longer than the α-queue; its length is determined by the packet loss requirements of the system. All packets leave the system via the α-queue.

When a packet enters the system a decision is made as to whether it should be placed in the α-queue or the β-queue. The packet is usually placed in the α queue. However, if the queue length of the α-queue exceeds some threshold $T_{onset,1}$ and it is decided that the traffic from this session contributes to the possibility of the α-queue overflowing, this packet and subsequent

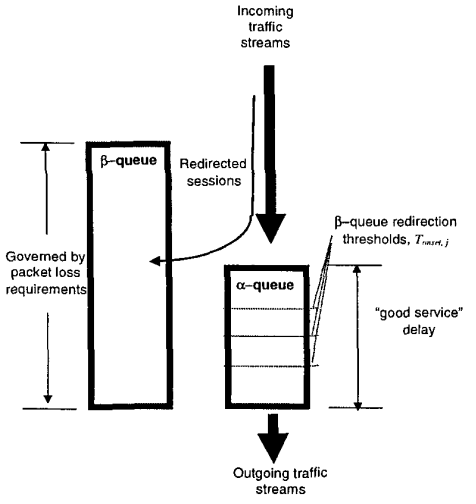

Fig. 1. The Dual Queue Scheduling Discipline: Packet transfer from α- to β-queue.

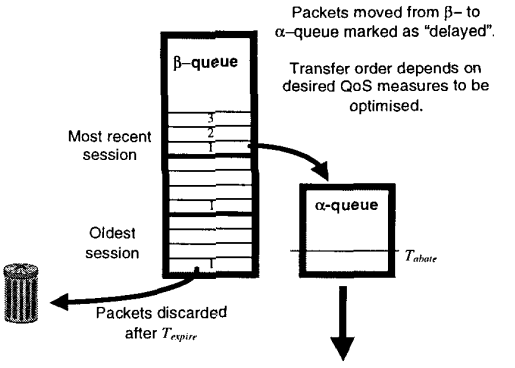

Fig. 2. The Dual Queue Scheduling Discipline: Packet transfer from β- to α-queue.

packets from this session are time stamped and placed in the β-queue. As subsequent thresholds, $T_{onset,j}$, $j > 1$, are crossed, further sessions are also directed to the β-queue.

Selecting sessions for the β-queue: There is a wide range of possible algorithms for selecting the sessions to be transferred to the β-queue depending on the properties desired. These include

- Random: Choose a session randomly from those currently in progress. Such a policy attempts to spread performance degradation across all sessions over their lifetime (but not instantaneously).
- Fair: Choose the session with the most packets in the α-queue at the time the threshold is crossed. Such a policy attempts to penalize those sessions whose instantaneous bandwidth requirement is higher than the fair share. This is on the grounds that it attempts to be instantaneously fair in deciding which sessions to penalize, but has the beneficial side-effect of minimizing the number of sessions simultaneously affected.
- Next packet: Choose the session of the packet which first breaks the next onset threshold. not already in the β-queue. Such a policy is a mix between being fair (the next packet arriving is likely to be from the session with the highest instantaneous bandwidth) and randomizing the sessions to be penalized. It has the added advantage of being the simplest algorithm to implement.
- Historical: Select a session that has been in the β-queue at some stage in the past, but is now back in the α-queue. Such a policy attempts to keep penalizing sessions that have already been penalized. It minimizes the amount of pain experienced by other sessions during their entire duration and thus attempts to maximize the number of users receiving good service over their entire session.

A timeout threshold, t_{expire} is set and all packets that have been in the queue longer than t_{expire} are discarded. This is especially important for real-time services, such as audio or video, which are not delay tolerant.

If the β-queue should overflow, the discipline may either drop the newly arriving packet, or push-out the oldest packet in the β-queue. The latter attempts to minimize the age of packets in the queue and thus maximize the chance that the transmitted packets will meet their end-to-end delay requirement.

Moving sessions back to the α-queue: When the length of the α-queue drops to the abatement threshold, T_{abate}, packets from the β-queue are moved back into the α-queue (see Figure 2). Packets from a session cease being redirected to the β-queue when there are no packets from that session left in the β-queue.

Again, there are a number of possible policies for deciding which packets to move first. Some of the policies are:

- FIFO: This is the simplest β-queue discipline and simply moves a packet to the α-queue in order of arrival when the length of the α-queue drops to T_{abate}. The number of packets from each session in the β-queue must be recorded so as to know when to stop redirecting a session.
- FIFO-FIFO: Each session transferred to the β-queue is maintained separately. When the α-queue drops below T_{abate}, packets from the first session placed in the β-queue are moved to the α-queue (that is, sessions are handled in a First In-First Out order). The packets from this session are given priority over other sessions contained in the β-queue. Within a session, packets are also transferred in First In-First Out order to preserve sequencing within a session.
- LIFO-FIFO: Each session transferred to the β-queue is maintained separately. When the α-queue drops below T_{abate}, packets from the session most recently placed in the β-queue are moved to the α-queue (that is, sessions are handled in a Last In-First Out fashion). The packets from this session are given priority over other sessions contained in the β-queue. Within a session, packets are still transferred in First In-First Out order to preserve sequencing within a session. Such an algorithm attempts to maximize the number of sessions receiving acceptable delay performance at any point time.
- Fair: Packets are transferred from the β-queue to the α-queue using a Fair Queueing approach when the length of the α-queue is less than or equal to T_{abate}. Thus, the sessions in the β-queue use the residual bandwidth left over by the sessions still using the α-queue. Such a policy attempts to be fair to those sessions currently being penalized.

Packets that have gone through the β-queue are marked to allow for identification by other nodes in the network. At subsequent nodes these will be the first choice for redirection to the β queue during a congestion episode. In this way, the network can co-ordinate which sessions are penalized if congestion occurs simultaneously at more than one point in the network.

IV. Simulation Setup

In this section we describe the simulation experiments. The simulation setup consists of a number of traffic sources generating MPEG data streams multiplexed onto a single transmission link towards a single destination.

A. MPEG Sources

Frame sizes from real MPEG traces generated by Rose [12] are used for sources in the simulation. Table I summarizes the characteristics of the MPEG sources. All sources have a peak rate of 4 Mbps at 25 frames per second and are 1600 seconds long (except for the *news1* trace which is 1260 seconds). They are randomly started over a 1600 s interval and measurements taken between 1600 s and 3200 s. When an MPEG trace comes to the end it is cycled.

The frames are broken into 1500 byte packets for transmission, with a 40 byte minimum packet size.

Number	*Name*	*Mean bits /frame*	*Peak /Mean Frame Size*	*Number of Frames*
1	asterix	22,349	6.6	40000
2	apt	21,890	8.7	40000
3	bond	24,308	10.1	40000
4	dino	13,078	9.1	40000
5	lambs	7,312	18.4	40000
6	movie	14,288	12.1	40000
7	mrbean	17,647	13.0	40000
8	mtv_1	24,604	9.3	40000
9	mtv_2	19,780	12.7	40000
10	news1	20,664	9.4	31515
11	news2	15,358	12.3	40000
12	race	30,749	6.6	40000
13	sbowl	23,506	6.0	40000
14	simpsons	18,576	12.9	40000
15	soccer1	27,129	6.9	40000
16	soccer2	25,110	7.6	40000
17	starwars	9,313	13.4	40000
18	talk1	14,537	7.3	40000
19	talk2	17,914	7.4	40000
20	terminator	10,905	7.3	40000

TABLE I
TRAFFIC CHARACTERISTICS OF THE MPEG TRACES USED AS TRAFFIC SOURCES [12].

B. Measurements

The following measurements are collected for each session:

- End to end delay for each packet
- Actual packets lost
- Degraded packets (packets delayed > 50 ms)
- Degraded frames (frames with degraded or lost packets)
- Degraded seconds (one second intervals with degraded frames).

We say that a session is receiving "good" service during any particular second if no frames of the session have been degraded. Any session not receiving *good service* is deemed to be receiving *degraded service*.

Experiment	*Number of sources*	*Offered Load (Mbps)*	*MPEG sources used*	*Buffer Space (Bytes)*	*Node bit rate (Mbps)*
1	20	9.5	all 20	500×10^3	8, 9, 10, and 12
2	4	1.6	1,2,3 & 5	100×10^3	1.3, 1.65, and 2
3	40	19	all twice	1×10^6	16, 20, and 24
4	40	30.7	race	1.5×10^6	24, 30, and 36

TABLE II
SIMULATION EXPERIMENTS

C. Dual Queue Parameters

The DQ algorithm implemented in these simulation experiments tries to give as many sessions as possible good service during a particular transient congestion episode. The β-queue follows the LIFO-FIFO discipline described in Section III. Packets from the β-queue are transferred to the α-queue only when the α-queue is empty, $T_{abate} = 0$. The thresholds for redirecting packets to the β-queue, $T_{onset,j}$ are set at: $L - L/(3+j-1)$ for $j : 1, \ldots, N-1$, where N is the number of sessions in the simulation and L is the length of the α-queue. Sources are removed from the redirection list when there are no packets from that session in the β-queue. In a real network with a changing number of sessions in progress, the threshold assignment can be made dynamic by simply replacing N with the number of session in progress.

The packet timeout parameter, t_{expire} is set to 400ms.

In this set of experiments we have chosen the simplest method of selecting which sessions are redirected to the β-queue (see Section III). We simply choose the session of the first packet to break $T_{onset,j}$ to redirect to the β-queue.

The β-queue uses an oldest packet push out mechanism on overflow.

V. Results

The set of simulation experiments outlined in table II are used to compare the performance of the Dual Queue (DQ) with a standard FIFO and Fair Queuing. Shreedhar and Varghese's Deficit Round Robin (DRR) technique is used for the Fair Queuing comparison [9], but enhanced to include old packet discard (DRRexp) to make a fairer comparison with DQ.

As noted earlier, we call a packet degraded if it is delayed more than 50 ms. We use this as an estimate of the maximum delay allowed before a frame cannot be displayed to the viewer. Packets that have been queued longer than t_{expire} = 400ms are discarded in the DRRexp and DQ algorithms. However, an MPEG frame that arrives too late for display may still be useful for the decoding of future frames which may not be so badly delayed.

All queues are given the same total buffer size. This is kept constant for the bit rates used in each simulation experiment. The DQ's α-queue size is changed whenever the link transmission speed is changed so as to give a maximum packet delay of 50 ms. It should also be noted that the worst case service delay in DRR introduced by the round robin scheduling is less than the good service delay threshold of 50 ms.

A. *Quality of Service Properties*

In experiment 1, all 20 heterogeneous MPEG sources are multiplexed onto the link. We have arbitrarily selected four representative sessions (3, 10, 14 and 20) to give a detailed comparison between the FIFO, DRRexp, and DQ scheduling algorithms.

Figure 3 shows the cumulative probability distributions of packet delay for these four sessions when the link transmission rate is 10 Mbps (compared with the total average offered rate of 9.5 Mbps). Note that we consider dropped packets to have infinite delay in the figure. We have chosen a highly loaded case in order to exaggerate the difference between the algorithms during congestion. A line is drawn at the 50 ms good service cut off point. We see that the DQ algorithm provides the highest fraction of packets receiving a delay less than 50 ms. This is achieved by giving some packets a delay much greater than 50 ms. Given that the service provided to such packets is already considered degraded, how much over 50 ms they are delayed is irrelevant. Note that DRRexp gives better service to session 20 than the other sessions as it is usually operates at less than its fair share of the bandwidth, as expected from a fair queueing based discipline. Note that is also true of the DQ discipline.

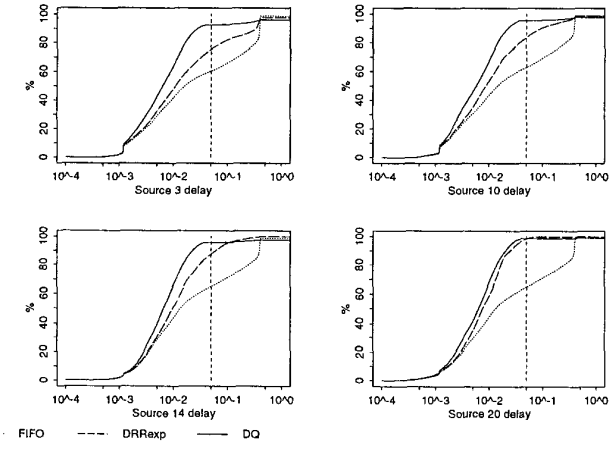

Fig. 3. Experiment 1, 10 Mbps: CDF of packet delays for 4 of 20 sessions. Dropped packets are considered to have infinite delay.

Figure 4 shows a plot of the highest packet delay per second for the same four sources when the link transmission rate is increased to 12 Mbps. Once again a line is drawn at the 50 ms mark to indicate the good service cut off point. The time range shown is chosen so as to include a number of transient congestion episodes for comparison. The FIFO is fair at the packet level. All sessions' packets, on average, experience the same delay passing through the FIFO. The DRRexp performs better, but fails to deliver good service when the bandwidth has to be shared in such a way that many of the sessions receive less than their required service. The priority nature of the Fair Queue can cause longer delays to a burst of packets than encountered in a FIFO. Thus, these extra delays can cause packets from a particular session to become degraded even when the link is not congested, as illustrated in the top trace. The DQ discipline selected tries to maximize the instantaneous number of sessions receiving good service so we see that one of the sessions receives very poor delay performance so that the others are spared service degradation at time 1925 and another at time 1975.

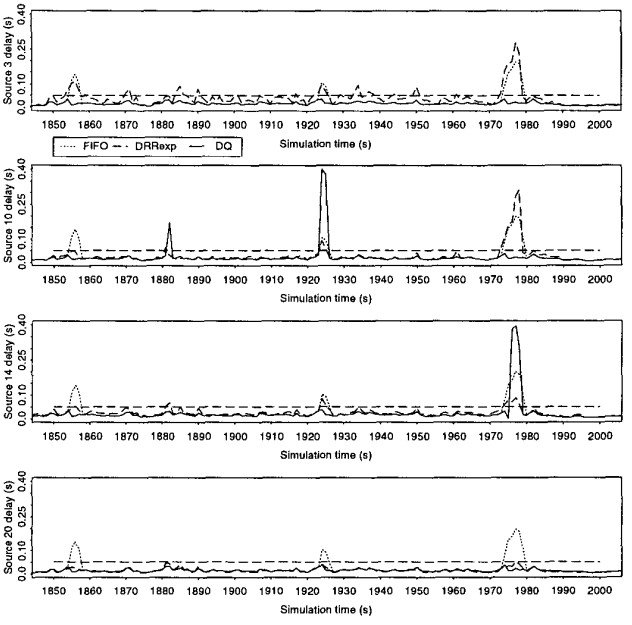

Fig. 4. Experiment 1, 12 Mbps: Maximum Packet Delay each second for 4 of 20 sessions.

This point is better illustrated by examining the QoS of all sessions, rather than just the four illustrated so far. Figure 5 shows the percentage of all sessions simultaneously receiving good services during the same period.

Recall that our main metric for QoS during any second is binary; service is either good or degraded. Hence, a binary representation of service per session is all that is required for comparisons between the scheduling disciplines. Figure 6 shows such a representation over the full measurement time scale for each session with a link transmission rate of 12Mbps. A vertical bar is drawn for every degraded second experienced by a session. The more white space on the graph, the better the service a session is receiving. All subsequent results will use this vehicle for comparison. We see that the DQ approach results in fewer sessions being affected by congestion than DRRexp at any time.

As the transmission rate is reduced, increasing the link utilization, the contrast between DRRexp and DQ becomes more vivid (Figures 7 and 8). This is where the real strength of the DQ approach becomes apparent as the DQ algorithm can provide good service to most of the sessions even under extreme overload conditions. It is also worth recalling that in this paper,

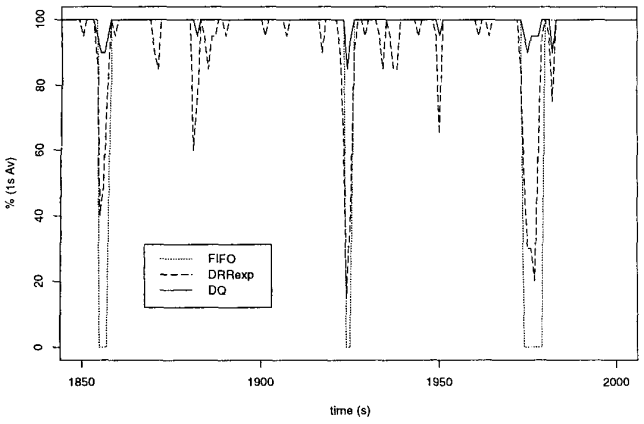

Fig. 5. Experiment 1, Link Transmission Rate 12 Mbps: Percentage of sessions receiving only good service in a 1 second period

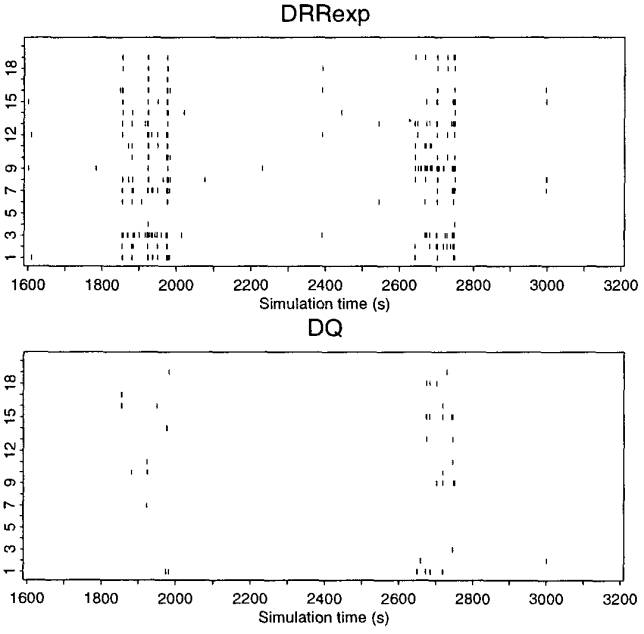

Fig. 6. Experiment 1, Link Transmission Rate 12Mbps: Degraded seconds for each session

we have studied the simplest possible implementation of the DQ approach, yet we are able to provide significantly better performance for real-time services than DRRexp.

B. Scalability and Fairness Properties

Experiments 2 and 3, which use 4 and 40 sources respectively, were conducted to examine the scalability of the DQ algorithm. Experiment 2 has the added complexity that the peak bit rates of any of the sessions is higher than the link transmission rate and hence will cause the α-queue to begin to fill. Looking at Figure 9 (Experiment 2, 4 sources), we see that even when presented with such an unmanageable load, the DQ performs as well as the DRRexp, if not slightly better, especially for sources 3 and 4. For Experiment 3 with 40 sources, the results are very similar to those shown for Experiment 1. The DQ performs better than

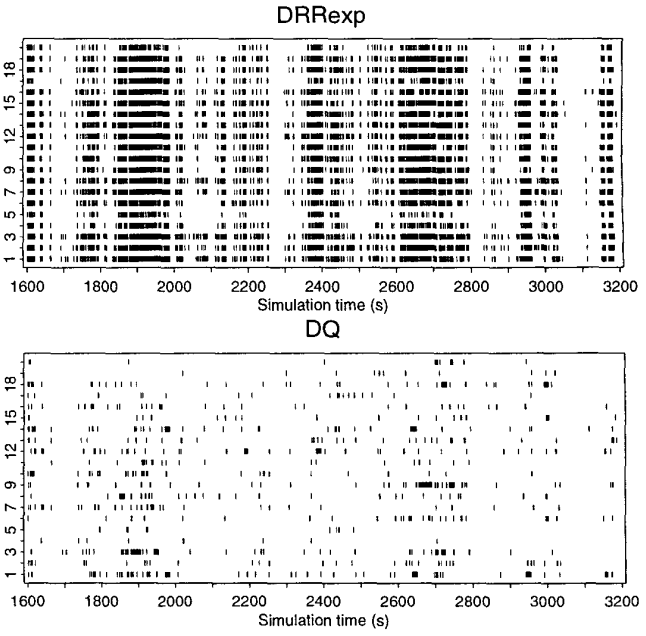

Fig. 7. Experiment 1, Link Transmission Rate 10Mbps: Degraded seconds for each session

DRRexp at the 3 loadings tested. As with experiment 1, higher loads show a greater contrast in their performance.

To examine fairness over moderate time scale, we use 20 homogeneous sources in Experiment 4. The sessions consist of 20 copies of the *race* trace started at random times over the period 0 to 1600 seconds. Figure 10 shows the degraded seconds plot for each session. We see that DRRexp provides fair, although degraded service, to each session. Qualitatively, the DQ is also providing fair service to each session *over the length of the simulation* rather than instantaneously. As seen earlier, the overall QoS of each session is significantly better than provided by DRRexp.

C. Summary

When there is no congestion all of the schemes compared give good service. As the periods of congestion increase, a greater contrast can be seen. Figure 11 shows the percentage of degraded packets for the FIFO, DRR, DRRexp and DQ algorithms for different average offered loads (normalized to the link transmission rate). Apart from achieving its aim of giving good service to as many sessions as possible during transient congestion periods, the DQ algorithm also minimizes the overall proportion of degraded packets. Note that the percentage of degraded under even this simple implementation of the DQ approach is close to the excess offered load above one Erlang, and hence is near optimal. While this would also be true for a short FIFO queue, the short FIFO has the disadvantage of resulting in unnecessary and indiscriminate loss during transient congestion when the offered load is close to, but less than the link speed.

During transient congestion episodes packets entering a normal FIFO queue all experience similar degraded service. If the congestion is caused by a small number of sessions, fair queuing

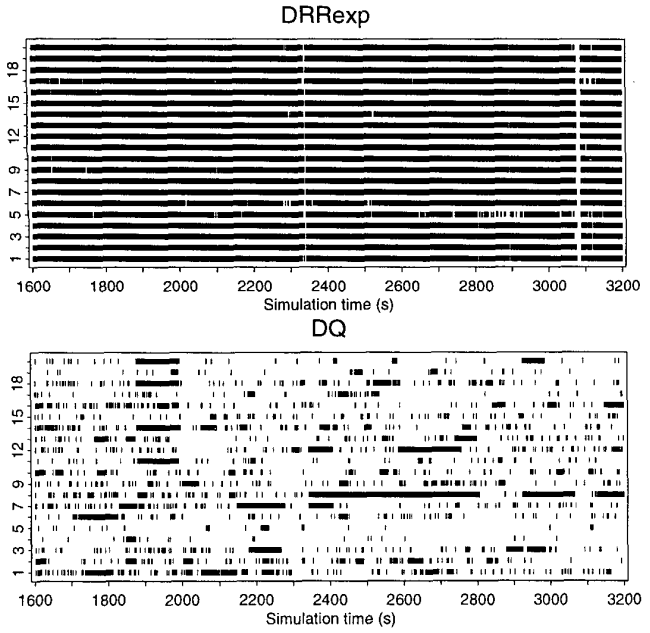

Fig. 8. Experiment 1, Link Transmission Rate 8Mbps: Degraded seconds for each session

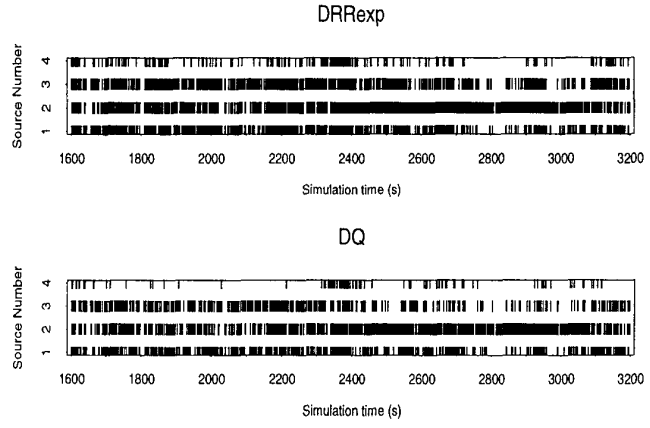

Fig. 9. Experiment 2, Link Transmission Rate 2Mbps: Degraded seconds for each session

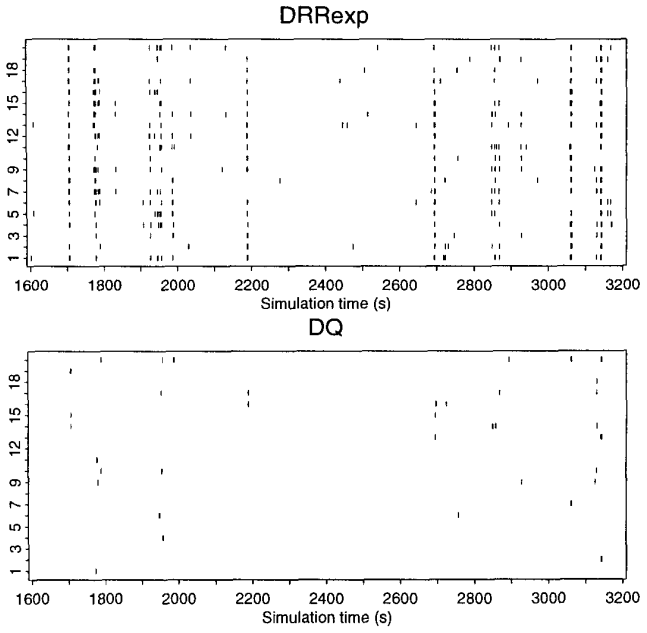

Fig. 10. Experiment 4, Link Transmission Rate 36Mbps: Degraded seconds for each session

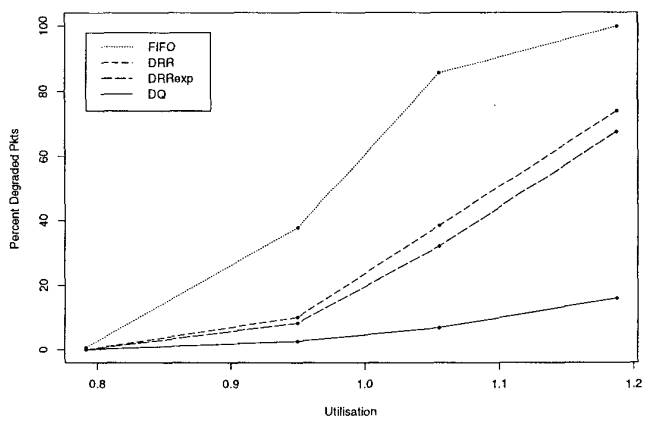

Fig. 11. Experiment 1: Overall percent of degraded packets versus normalized utilization

provides relief for the other sessions. However, results presented here indicate that this is not always the case, as transient congestion is often caused by many sessions whose combined traffic fills the queue. Fair queuing divides the bandwidth equally, thus degrading the service of a large number of sessions, and thus a large number of packets. The DQ algorithm outperforms fair queuing because it degrades the service of only enough sessions so as to obtain good service for the rest during the congestion episode, thus also increasing the good-put.

VI. Discussion

By relaxing the requirement of instantaneous fairness stressed in FQ based approaches, we are able to optimize other measures of service quality, and trade different measures off against each other. As mentioned in Section III, the Dual Queue structure has the flexibility to implement a wide range of scheduling paradigms. The results presented in the previous section show that the Dual Queue can provide improved overall quality but retain moderate fairness on the time-scale of seconds. However, there are other options.

For example, if users complain every time they make a connection and receive unsatisfactory QoS, the service provider may wish to minimize the number of complaints received. In this case, the service provider may choose to maximize the number of users receiving "perfect" service, that is, showing no sign of congestion. This can be achieved by recording in a "target" list the connections which have been redirected to the β-queue at some stage, and increasing the probability that they will be redirected again the next time the system is congested. Thus the degradation is not spread evenly among users, even over con-

siderable time scales, increasing the probability of a connection remaining unaffected by the congestion.

The probability of redirecting connections in the target list may be increased to 1, so that no new connection will be redirected until all connections which have previously been redirected are currently being redirected. Another alternative is to consider several factors, including the current data rate as well as the presence in the target list.

If the aim is to maximize the number of satisfied customers at any moment in time, the optimal strategy is to redirect the highest rate sessions. If the statistics of a given connection change significantly with time, the above approach of repeatedly targeting a particular user for the entire connection may not be appropriate. Instead, connections could be removed from the target list after some time, t_{target}. This would result in bursts of bad service on a time scale of t_{target}, but may improve the total number of customer-minutes of high QoS.

Alternatively, a telecommunications regulatory body may require a stricter guarantee of fairness over moderate time scales than that provided by the Dual Queue implementation studied in this paper. This is also possible, again by recording those connections which have been redirected. This time, these connections will be *less* likely to be redirected again. Once again, there is a trade-off with the total number of customer-minutes of high QoS which can be achieved.

Even greater flexibility is possible if other service disciplines are considered for the β-queue, rather than only the LIFO-FIFO studied in Section III. For example, the β-queue could itself use fair queueing (see section III). Note that if the size of the α-queue is set to zero, then the hybrid approach reduces to fair queueing.

There is also a wide variety of possible trigger mechanisms other than simple threshold crossing based schemes for initiating the move of sessions to the β-queue. Possible examples include use of derivative based schemes in which the rate of growth of α-queue is used, or fuzzy logic based approaches.

The foregoing indicates some of the many policies which can be implemented using the Dual Queue structure. We are not advocating one to the exclusion of the others, as each seeks to meet different requirements. Our aims are to point out that focusing on instantaneous fairness reduces many other measures of service quality, which may be of greater interest to both the user and service provider, and to propose a more flexible scheduling approach allowing a tradeoff between QoS measures.

VII. Conclusions

In this paper we have examined issues related to the definition of Quality of Service for real-time packet based services such as multimedia data, MPEG video streaming and IP telephony.

We have shown that there are a number of potential Quality of Service criteria not supported by existing real-time scheduling disciplines, such as FIFO and Fair Queueing. Specific examples of such criteria include maximizing the proportion of customers receiving good service at any particular moment, or indeed, over the lifetime of their session.

To overcome this deficiency, we have proposed a new approach to real-time scheduling, which we term the Dual Queue Approach. The Dual Queue has the flexibility to satisfy a variety of QoS objectives, ranging from existing notions of fairness through to maximizing the number of customers receiving good service.

We have shown that even the simplest Dual Queue provides excellent service to many more customers simultaneously than other disciplines. Moreover, it is scalable in the number of active sessions, and can be made fair, if desired, over moderate to long time scales.

Acknowledgments

The first and second authors gratefully acknowledge the funding support of Ericsson Australia, Pty Ltd.

References

[1] Special issue on real-time video services in multimedia networks, *IEEE J. on Selected Areas in Commun.*, vol. 15, Aug. 1997.

[2] L. Zhang, "VirtualClock: A new traffic control algorithm for packet switching networks," *ACM Trans. Comp. Sys.*, vol. 9, pp. 101–124, May 1991.

[3] S. Golestani, "A self-clocked fair queuing scheme for broadband applications," in *Proc. IEEE INFOCOM '94*, 1994, pp. 636–646.

[4] C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time communication in packet-switched networks," *Proc. IEEE*, vol. 82, pp. 122–139, Jan. 1994.

[5] J. Liebeherr, D. W. Wrege, and D. Ferrari, "Exact admission control for networks with a bounded delay service," *IEEE/ACM Trans. Networking*, vol. 4, pp. 885–901, Dec. 1996.

[6] H. Zhang, "Service disciplines for guaranteed performance service in packet switching networks," *Proc. IEEE*, vol. 83, pp. 1374–1396, Oct. 1995.

[7] R. Mazumdar, L. G. Mason and C. Douligeris, "Fairness in network optimal flow control: optimality of product forms," *IEEE Trans. Commun.*, vol. 39, pp. 775–782, 1991.

[8] A. Varma and D. Stiliadis, "Hardware implementation of fair queuing algorithms for asynchronous transfer mode networks," *IEEE Communications Magazine*, vol. 35, pp. 54–68, Dec. 1997.

[9] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round robin," *IEEE/ACM Trans. Networking*, vol. 4, no. 3, 1996, pp. 375–385.

[10] A. Demers, S. Keshav, S. Shenker, "Analysis and simulation of a fair queueing algorithm," *Proc. SIGCOMM '89*, 1989.

[11] D. Hayes, L. Andrew, M. Rumsewicz, "Dual Queue Approach to Improving Network Performance During Transient Congestion Episodes," *Proceedings of IEEE LAN/MAN Workshop 98*, Banff, Canada, May 1998, pp. 51–56.

[12] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," in *Proceedings of the 20th annual conference on local computer networks*, 1995, pp. 397–406. MPEG traces at ftp://ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG.