WM4-2 2:20

Proceedings of the 1999 IEEE
International Conference on Control Applications
Kohala Coast-Island of Hawai'i, Hawai'i, USA • August 22-27, 1999

Impact of Flow Control on Quality of Service Driven Packet Scheduling Disciplines.

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 investigate the interaction between window based flow control and a recently proposed packet scheduling discipline designed for real-time services. The scheduling discipline, called the Dual Queue discipline, has been shown to provide greater flexibility than other scheduling disciplines such as fair queueing. However, its performance on non real-time services has not been previously investigated. We show that the Dual Queue's performance for non real-time services is again better than that of alternative approaches, which indicates that it will perform well in an environment of mixed real-time and non real-time traffic, such as the current internet.

Key words: Quality of Service, packet scheduling, Dual Queue, TCP.

1 Introduction

There is an increasing trend to mix real-time services with non real-time services on a single network, in particular, the internet. Real-time services such as telephony, video and virtual reality applications [1, 2, 3] are being combined with non real-time services such as World Wide Web (WWW) traffic and bulk file transfers. Such networks must provide sufficient Quality of Service (QoS) to both types of sessions with respect to several quality measures, such as effective throughput, packet delay and loss rates. Guarantees with respect to QoS for real-time services require effective real-time scheduling. However, such scheduling should not impact adversely on the reliability or throughput of non real-time services sharing the network.

In [4, 5] the Dual Queue (DQ) scheduling algorithm was proposed. The DQ algorithm was designed for congestion management of "soft" real-time services [6], such as multimedia data, MPEG video streaming and IP telephony, which can tolerate a small degree of loss or delay. For such services, the Dual Queue has been shown to provide better service than either First In First Out (FIFO) scheduling or Fair Queueing.

The DQ scheduling algorithm was designed to fill a similar role to fair queueing. It provides an infrastructure that allows selectable QoS measures, including maximising the number of users receiving good service. It achieves 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, it has been shown [4, 5] that fairness can be attained over time scales of the order of seconds, and still provide substantially better overall quality of service for real-time services than Deficit Round Robin (DRR) [7], a practical implementation of a Fair Queueing [8] discipline.

The DQ's performance has not been investigated for reliable data communications, in which lost packets are retransmitted, and flow control is used to reduce the transmission rate during periods of congestion. A typical example of such protocols is the Transmission Control Protocol (TCP), which is used in the WWW and is fast becoming the dominant protocol for wide area packet switched networks. Examining this question is critical to determining whether the DQ may be suitable for mixed environments.

In this paper we examine the impact of the DQ on non real-time services. We show that the DQ scheduling discipline

- attempts to maximise the number of sessions receiving good service.
- achieves better effective throughput than FIFO and DRR for bursty loads.
- achieves a better average session bit rate than FIFO and DRR for bursty loads.
- performs marginally better than FIFO and DRR for a continuous load.

The rest of this paper is organised as follows. A description of the Dual Queue scheduling discipline is given in Section 2. The issues arising when flow control is introduced are outlined in Section 3. Section 4 describes the simulation scenario used to determine the performance of the al-

0-7803-5446-X/99 $10.00 © 1999 IEEE

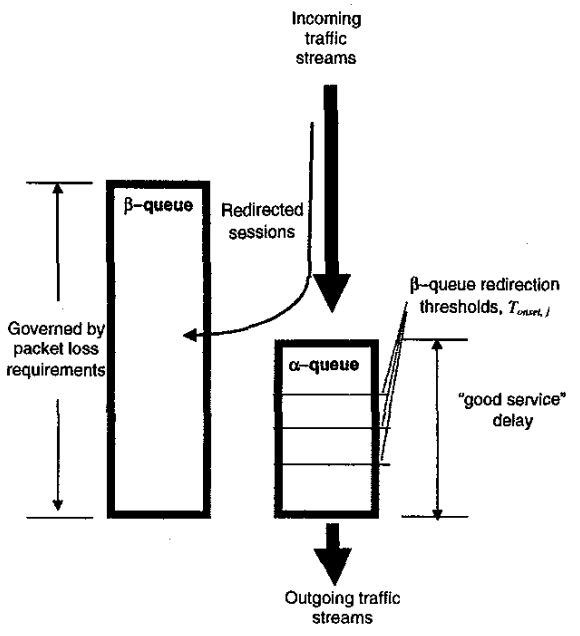

Figure 1: The Dual Queue Scheduling Discipline: Packet transfer from α to β queue.

gorithm, and simulation results are presented in Section 5. Conclusions are presented in Section 6.

2 The Dual Queue Scheduling Discipline

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

The α queue is a short FIFO queue. Its length corresponds to the delay acceptable by real-time services. The β queue may be significantly longer than the α queue; its length is determined by the packet loss requirements of the system, and ensures that the number of packet retransmissions is not excessive.

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

A timeout threshold, t_{expire}, is set and all packets that have been in the queue longer than t_{expire} are discarded. For non

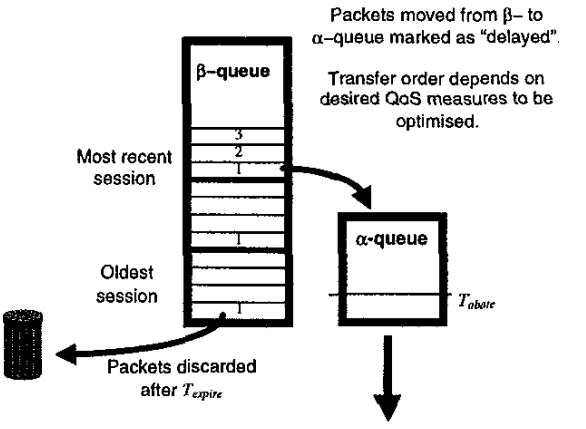

Figure 2: The Dual Queue Scheduling Discipline: Packet transfer from β to α queue.

real-time services, this is determined by the retransmission timeouts used by the higher layer protocols. For real-time services, t_{expire} is determined by the maximum age a packet can still be of some use in the decoding process.

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 minimise the age of packets in the queue and thus maximise the chance that the transmitted packets will meet their end-to-end delay requirement. However, if go-back-N retransmission is used, discarding the most recently arriving packets is preferable.

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 many alternative strategies for moving packets out of the β queue which give the discipline different properties.

It was demonstrated in [4, 5] that the DQ algorithm attempts to minimises the number of MPEG sessions that receive degraded service during a transient congestion episode. In a real-time video application, MPEG frames delayed more than a certain amount are unable to be used for displaying the video. Thus during a transient congestion episode, the DQ scheduling discipline selectively degrades only enough sessions, such that the remainder will receive good service. The FIFO, and to a lesser extent the DRR, disciplines delay frames from all sessions, thus degrading all sessions during the transient congestion episode.

3 Issues arising with flow control

Reliable non real-time services employing flow control are fundamentally different from real-time services, in two principal ways.

Firstly, congestion can be relieved by causing sources to slow their rate of transmission, rather than simply discarding packets. Although TCP networks usually do not directly control the source rates, delaying packets will cause the flow control algorithm to reduce its rate automatically. Delaying packets of some sources more than others will cause some sources to be unaffected by the congestion, which was the aim of the Dual Queue approach.

The other difference is that discarded or delayed packets must be retransmitted. When a real-time packet is discarded, the penalty is simply the lost data. In contrast, when non real-time packets are discarded, the congestion episode is prolonged by the retransmissions, and it is in principle possible to have an unstable system where such retransmissions dominate traffic, and the throughput tends to zero.

4 Simulation setup

The aim of the simulation study is to compare the performances of the FIFO and DRRexp disciplines with that of the DQ discipline under TCP loads. Performance is measured in terms of: average session bit rate, effective node throughput, and degraded sessions.

The simulation setup is shown in Figure 3 and consists of a number of traffic sources generating TCP data streams multiplexed onto a single transmission link.

4.1 Traffic Sources

The traffic from each source consists of a sequence of bursts. The length of a burst is drawn from a Pareto distribution, given by

$$p(x) = 1 - \left(\frac{a}{x}\right)^c \qquad (1)$$

for $x \geq a$, where $1 < c < 2$, and the mean value is $ca/(c-1)$. This has been shown to be a reasonable model for actual data burst sizes [9]. The time between the completion of one burst and the start of the next is given by an exponential distribution. This inter-burst time represents a user's "think time" in an interactive session. Each burst is a separate TCP session, complete with slow start, as would be the case for WWW browsing.

4.2 TCP Implementation and Parameters

The TCP implementation generally follows IETF RFC 793, but uses the Jacobson/Karels algorithm [10] for estimating the packet round trip time and calculating the timeout for each transmitted packet. This simulation uses a high resolution clock for round trip and timeout calculations. The parameters are according to the recommendations in [10]. The slow start congestion avoidance algorithm is used whenever the link has been idle, whether due to a new session start or a wait for a timeout.

The simulation has a fixed packet return path delay, sink to source, of 500 ms.

4.3 Dual Queue Parameters

The DQ algorithm implemented in these simulation experiments tries to give as many sessions good service as possible. Good service is defined in terms of delay: a packet is deemed to have received good service if its delay is under 120 ms. Therefore a session whose packets are delayed less than 120 ms is considered to be receiving good service during the period being measured. The DQ's α queue size has been set to give a maximum packet delay of 50 ms, which corresponds to the value used in [4, 5] for real-time traffic. So in effect it aims to give as many sessions as possible a delay of less than 50 ms, well below the "good service" delay that has been set for non real-time traffic in this simulation.

There are several possible policies for deciding which packets to move first. This paper uses a LIFO-FIFO discipline. Each session transferred to the β queue is maintained separately. When the length of the α queue drops to T_{abate}, packets from the session most recently placed in the β queue are moved to the α queue (last in-first out by session). Within a session, packets are still transferred in FIFO order. The congestion abatement threshold was $T_{abate} = 0$.

The congestion onset thresholds are $T_{onset,j} = L - L/(2 + j)$ for $j = 1,2,\ldots$, where 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. The jth session to be redirected to the β queue is chosen by a simple procedure which identifies "busy" sources. When a packet from source i causes the α queue length to exceed $T_{onset,j}$, the scheduler checks how many packets from source i are currently in the α queue. If that number exceeds a small threshold θ_j, then source i is redirected to the β queue. If not, then the packet is stored in the α queue as normal, and θ_j is decreased by 1, to increase the probability that the next packet will cause a source to be redirected. Once a source has been redirected, θ_j is reset to its original value. Here $\theta_j = \theta + 1 - j$ for $j = 1,2,\ldots\theta$. Here $\theta = 5$.

The packet timeout parameter, t_{expire} is set to 5s. The β queue uses an oldest packet push out mechanism on overflow.

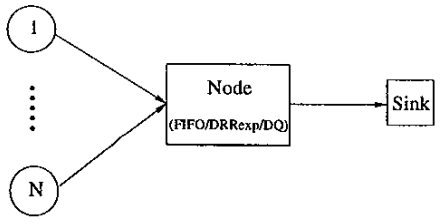

Figure 3: Simulation Set Up

Table 1: Simulation Experiments

Experiment Number	*Pareto Parameters (a,c)*	*Average Silence Times (s)*	*Buffer size (Bytes)*	*Number of Sources*	*Source Bit Rates*	*Node Bit Rate*
1	6923, 1.3 (mean of 3×10^4 bytes)	0.5, 1, 1.5, 2, 2.5, 3, 5, 7.5, 10, and 12.5	500000	50	500 kbps	2 Mbps
2	231×10^6, 1.3 (mean of 1×10^9 bytes)	0.01	500000	50	500 kbps	2 Mbps

5 Results

The set of simulation experiments outlined in Table 1 is used to compare the performance of the DQ with a standard FIFO and Shreedhar and Varghese's DRR Fair Queueing technique [7]. DRR was enhanced to include old packet discard (DRRexp) to make a fairer comparison with DQ. Packets that have been queued longer than t_{expire} = 5s are discarded in the DRRexp and DQ algorithms.

All queues are given the same total buffer size. 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 120 ms.

5.1 Bursty load

Experiment 1 compares the different scheduling disciplines under differing loads. The simulation models World Wide Web (WWW) type traffic. Users select a page, load and read it, and then select another. The load is increased by reducing the "think time" (average silence time).

Figure 4 shows a typical plot of the operation of the DQ discipline compared with FIFO and DRRexp in terms of times the sessions were degraded. Only 10 of the 50 sources are shown for clarity. Shading marks the sessions, while blacked out sections illustrate the times degraded service is being received. The DQ discipline attempts to give as many sessions as possible good service during a congestion episode. Looking at a particular congestion episode, say at simulation time equals 70 s, it is seen that the DQ ensures relatively fewer sources receive degraded service than the other scheduling disciplines.

Practically the DQ's operation, as described above, changes the way a user will experience WWW browsing. Many users will not experience a particular transient congestion episode, however those that do will experience much longer delays than would be the case for the other scheduling disciplines. With the current DQ parameters this perception of quality of service will change from page to page. Changing the DQ parameters can keep the perception of service more uniform from page to page.

Users are most interested in how long they must wait for a particular operation, such as a file download, rather than lower level measures such as packet loss rate before retransmission or network throughput. We measure the average bit rate experienced by each connection (TCP burst), calculated as follows:

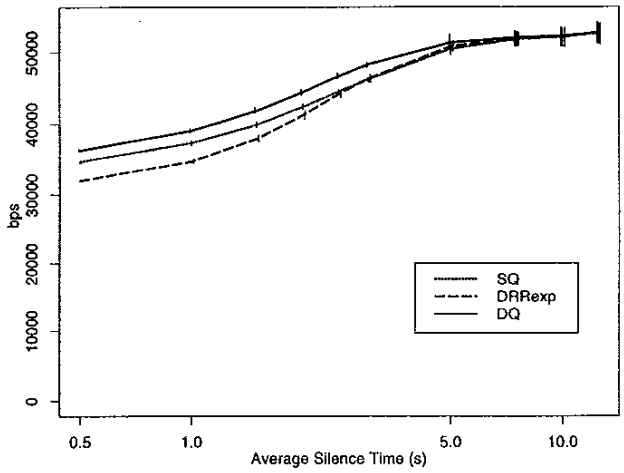

Figure 5: Mean data rate during transfer sessions as a function of average silence time between sessions.

$$\overline{bit\ rate} = \frac{\sum_{n=1}^{N}\sum_{s=1}^{S(n)} b_{n,s}}{\sum_{n=1}^{N}\sum_{s=1}^{S(n)} t_{n,s}}; \quad (2)$$

where $b_{n,s}$ is the number of valid bits received in a session, $t_{n,s}$ is the session time, N is the number of sources, and $S(n)$ is the number of sessions for source n. Figure 5 shows that the DQ discipline gives better performance than DRRexp and FIFO, particularly for higher loads (left end of the x-axis where the average silence time is smaller. Each point was calculated from 20 simulation runs, with the 95% confidence bars drawn. The DQ discipline had a higher mean data rate than the other disciplines in all individual runs.

Although users are interested in their own delays, operators are interested in their effective throughput(throughput of actual data excluding packet overheads and unnecessary retransmissions). All three schemes have very similar effective throughput except at high loads, where DQ is slightly better than the others (1.66 Mbps compared to 1.60 Mbps for FIFO and 1.50 Mbps for DRR, when the average "think time" is 0.5 s).

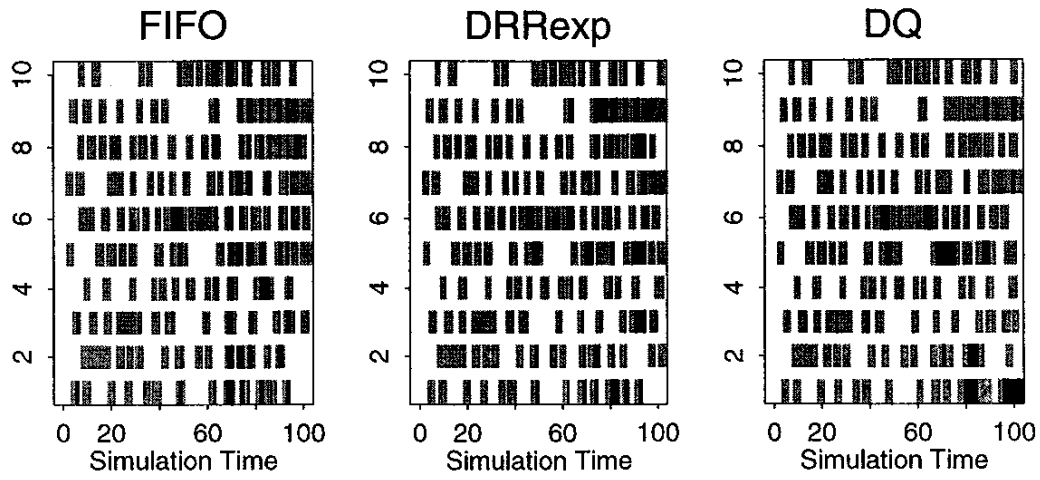

Figure 4: Degraded Portions of Sessions (3 s "think time")

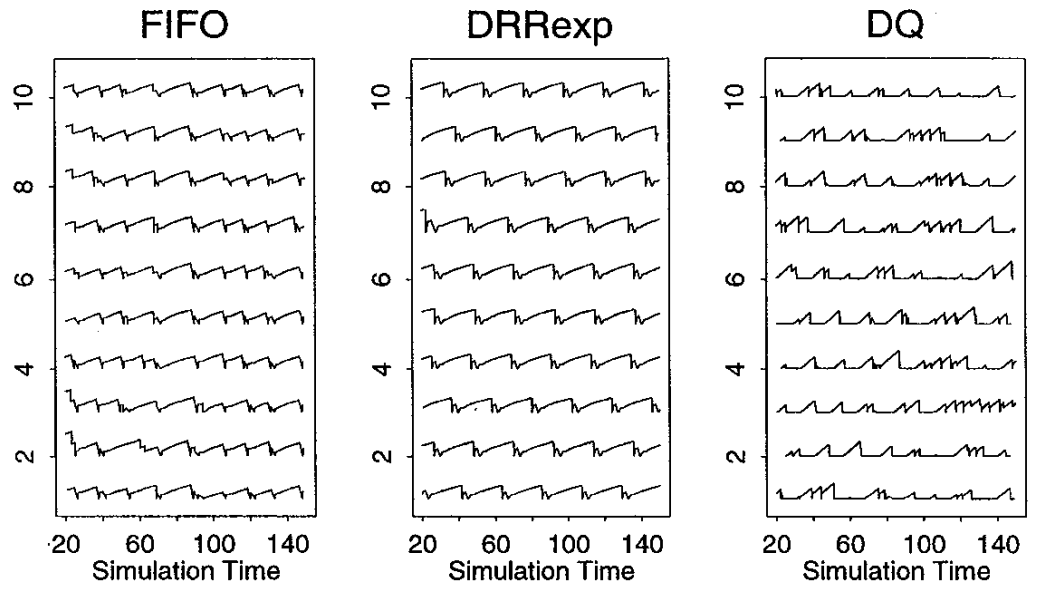

Figure 6: TCP window size against time.

5.2 Continuous load

It is important that scheduling disciplines perform well under a range of conditions. We now examine a continuous load, in which each user constantly has data to transmit. Experiment 2 analyses this condition by ensuring the minimum burst of data is 231×10^6 bytes. In this case, all three approaches have very similar effective throughput, but DQ is marginally better (1.68 Mbps DQ, 1.58 Mbps FIFO, 1.59 Mbps DRR).

For maximum throughput, it is important that the TCP windows do not all shut down or open up simultaneously. TCP window sizes of different sources feeding into a FIFO queue can become synchronised in a quasi-periodic pattern [11]. This causes alternate periods of link starvation and packet loss/retransmission, which both lower the throughput. The decoupling effect of FQ causes the windows to follow identical but initially unsynchronised periodic patterns. Typical traces are shown in Figure 6. Only 10 of the 50 sources are shown for clarity. In contrast, the window sizes with DQ show little structure. This accounts for DQ's superior effective throughput.

It is also important to note is that there is continual turnover of the sessions in the β queue. If this were not the case then it would be possible to obtain high overall effective throughput while some sessions were receiving no service. The congestion control windows of sources in the α queue tend to start to synchronise and cycle since the α queue is a FIFO. As a result the α queue periodically drops to T_{abate} allowing β queue packets to be transmitted. Sources that have been redirected to the β queue have slowed their transmission rates as a result of their low throughput. Old packets are discarded, leaving relatively few packets in the β queue. The DQ rarely suffers from link starvation, as there are usually packets in the β queue to transmit when the α queue drops to T_{abate}, and the cycling of sources through the β queue disrupts the α queue synchronisation.

Appropriately setting the value t_{expire} can aid this process. If t_{expire} is greater than the maximum TCP packet timeout bandwidth can be wasted by unnecessarily transmitting duplicate packets. If t_{expire} significantly less than the maximum TCP packet timeout, useful packets may be unnecessarily discarded. This simulation uses a maximum TCP packet timeout of 5.5 s and t_{expire} equal to 5 s.

6 Conclusions

In this paper we have examined the interaction of flow control and retransmissions, such as provided by TCP, with the Dual Queue packet scheduling discipline which was designed to provide high quality of service for real-time services.

We have found that the performance of DQ, in terms of mean transfer rate and total effective throughput, no worse than that of the FIFO and DRRexp disciplines for moderate to low loads, and is significantly better than the other disciplines for high loads. Previous studies [4, 5] have demonstrated that the DQ is significantly better than alternatives in environments tuned to real-time applications. The results of [4, 5], combined with the findings presented here, suggest that DQ will be a suitable discipline for networks combining mixed real-time and non real-time services.

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] D. A. Hayes, L. Andrew, M. Rumsewicz, "Dual Queue approach to improving network performance during transient congestion episodes", *Proc. IEEE LAN/MAN Workshop 98*, Banff, Canada, May 1998, pp. 51–56.

[5] D. A. Hayes, M. Rumsewicz, L. Andrew, "Quality of service driven packet scheduling disciplines for real-time applications: Looking beyond fairness", in *Proc. IEEE INFOCOM '99*, 1999, vol. 1, pp. 405–412.

[6] 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.

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

[8] A. Demers, S. Keshav, S. Shenker, "Analysis and simulation of a fair queueing algorithm", *Proc. SIGCOMM '89*, 1989, pp 1–12.

[9] M. E. Crovella, A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes", *Proceedings of the 1996 ACM SIGMETRICS International Conference*, May 1996, pp. 160–169.

[10] L. L. Peterson and B. S. Davie, *Computer networks: A systems approach* Morgan Kaufman, San Francisco, 1996.

[11] K. Cheon and S. S. Panwar, "On intelligent cell discarding policies for TCP traffic over ATM-UBR", *Proc. IEEE LAN/MAN Workshop 98*, Banff, Canada, May 1998, pp. 371–383.