

# Scheduling disciplines for multimedia WLANs: Embedded round robin and wireless dual queue

Ravindra S. Ranasinghe\* Lachlan L.H. Andrew\* David A. Hayes\*\* David Everitt†

\* The ARC Special Research Centre for Ultra Broadband Information Networks (CUBIN),

\*\* Ericsson/Melbourne University Laboratory (EMUlab),  
Dept of Electrical and Electronic Engineering  
The University of Melbourne, VIC 3010, Australia  
{rran, l.andrew, d.hayes}@ee.mu.oz.au

†Centre for Convergent Technologies (CCT),  
School of Biophysical Sciences and Electrical Engineering,  
Swinburne University of Technology  
PO Box 218 Hawthorn, Victoria 3122, Australia  
deveritt@swin.edu.au

**Abstract**—Wireless local area networks have developed into a promising solution to support advanced data services in untethered environments. Selection of an efficient packet-scheduling scheme is important for managing the bandwidth while satisfying QoS requirements of active sessions having diverse traffic characteristics. The key difficulty is the distributed nature of the queues in the uplink, resulting in the scheduler having to trade off polling greedy stations against wasting resources by polling potentially idle stations. In order to address this, we propose a novel scheduling scheme, “Embedded Round Robin”, which dynamically classifies stations as “busy” and “clear”. We then extend the recently proposed Dual Queue scheduling discipline to the case of wireless networks.

## I. INTRODUCTION

Wireless Local Area Networks (WLANs) are increasingly in demand. In such networks, the limited bandwidth must be managed carefully (i.e. packet scheduling) to provide high quality real-time services [1].

Scheduling downlink packet streams is essentially a centralized task, and is analogous to scheduling in wired networks. Many centralised scheduling algorithms to achieve different quality-of-service (QoS) characteristics have been proposed. These characteristics include low delay [2], [3], low delay jitter [4], fairness [5], [6], [7], [8], [9], and maximum end user satisfaction [10]. All of these schemes require the scheduler to know the state of each active session (eg. packet arrival times, queue lengths, packet lengths etc.). These algorithms can be easily implemented in a centralised queuing environment since the required information is available.

The uplink of wireless networks presents particular difficulties for packet scheduling, especially in the presence of non-uniform load. When a distributed queuing system is controlled using a centralised scheduler, an obvious mechanism of granting channel access to mobile stations is polling. One possible strategy for managing the polling is round robin (RR) [11], which polls stations cyclically, regardless of the state of their queues. When RR is used to schedule service from a centralized queue, the scheduler can efficiently bypass stations that have no data to transmit. In contrast, the scheduler of a wireless network must poll a station in order to determine that it has no data to transmit. This overhead can be significant; for example, in IEEE 802.11 wireless LANs [12] it takes 5% of

the time of a maximum length packet transmission at a 1Mbps channel rate.

If the scheduler is able to poll queues at different polling rates then it can save bandwidth. Existing non-uniform polling schemes such as [13] require explicit messaging which not only introduces transmission overhead but is also incompatible with many WLAN standards such as 802.11 [12]. Other MAC protocols not based on polling [14], [15] have been proposed to extend to wireless networks some of the fair queuing and deadline-ordered scheduling schemes popular in wired networks. These again achieve QoS at the expense of explicit messaging. Ranasinghe et al. [16] have proposed the distributed deficit round robin scheme, a method of implementing a distributed version of the deficit round robin scheme [9] compatible with the proposed standards.

In this paper, we propose a novel round robin type scheduling discipline to generate the poll requests with varying inter-poll time depending on the load of each traffic source. In this embedded round robin (ERR) scheme detailed in Section II, the stations are dynamically classified as “busy” or “clear”. The scheduler services “busy” stations more frequently than “clear” ones. This reduces the polling overhead and consequently increases the bandwidth available for data.

Even with this increased bandwidth, there will be transient periods of congestion, when resources must be rationed. Despite the recent attention to fair queuing, it has been suggested that fairness may not be the best aim at such times [10]. Since the user is unable to observe the QoS received by other users and its relative fairness, it may be preferable to discard the notion of fairness and aim to maximise the number of sessions that receive acceptable service. This is achieved by letting a minority of overloaded streams bear the brunt of the degraded service. This frees resources to allow the number of satisfied users on the system to be maximised. The recently proposed Dual Queue (DQ) scheduling scheme [10] addressed this for centralised queuing systems. That scheme has the flexibility to satisfy a variety of QoS objectives ranging from existing notions of fairness through to maximising the number of contented users.

The DQ algorithm uses information which is unavailable in

the distributed wireless uplink. Section IV presents an implementation of the DQ philosophy based on the ERR polling strategy. This algorithm, which we call the Wireless Dual Queue (WDQ), again aims to maximise of the number of sessions that receive acceptable service.

The performance of the ERR and WDQ polling algorithms are compared with that of the simple RR algorithm for a small number of stations transmitting compressed video traffic. The results in Section V show that ERR outperforms the standard RR policy, and WDQ provides the best overall performance of the schemes tested.

## II. EMBEDDED ROUND ROBIN (ERR) SCHEME

As explained in Section I, one of the major problems associated with scheduling distributed uplink queues is the lack of information available to the scheduler. Therefore the scheduler may poll potentially idle stations, wasting bandwidth. If the scheduler has sufficient information regarding how busy the remote station was when the poll requests were received in the past, it can identify which stations were busier than others. This concept is the basis for the embedded round robin (ERR) scheme.

In ERR, stations known to have data to transmit are classified as “busy” while the remainder are classified as “clear”. Many polling protocols already provide a single feedback bit for this purpose. Importantly it is supplied by the IEEE 802.11 MAC header [12]. Therefore ERR can be implemented without any protocol changes.

The scheduler services clear stations in round robin order. However, between consecutive services of clear stations, it performs a complete round robin cycle of the busy stations. This is illustrated in Figure 1 where stations A, B and C are “busy” and stations D, E, F and G are “clear”. This reduces the time wasted on polling idle or low-rate stations. When there are only a few busy stations, they receive most of the bandwidth, hence their backlog is soon cleared and polling of clear stations is not unduly delayed.

When the number of busy stations increases, the backlog can remain for some time, and substantially increase the duration of a polling cycle. This could cause packets arriving at “clear” stations to receive unnecessarily poor service. To prevent this, the busy round robin cycle services may have to be preempted to give the scheduler attention on “clear” stations. In order to accomplish this, the scheduler can measure the total transmission time of the ongoing “busy” servicing episode and terminate the “busy” servicing if it exceeds a defined threshold. This algorithm is formally described in Algorithm 1.

## III. THE ORIGINAL DUAL QUEUE

As mentioned in Section I, the user-percieved QoS degrades during transient congestion periods. It is however possible with the dual queue (DQ) algorithm, which was proposed to combat transient congestion in routers, to shield a subset of users from it so that they are completely unaware of the congestion

```

WHILE (SchedulingList is not empty)
  i := next clear station in RR order
  Poll station i
  Receive the response p
  IF (p flags "more data")
    Mark station i as busy
  END-IF

  FOR k := 1 to number of busy stations
    bsyPtr := nextBusy(bsyPtr)
    Poll station bsyPtr
    Receive the response p
    IF (p flags "no more data")
      Mark station bsyPtr as clear
    END-IF
    IF (busytime exceeded)
      break from FOR loop
    END-IF
  END-FOR
END-WHILE

```

Algorithm 1. ERR

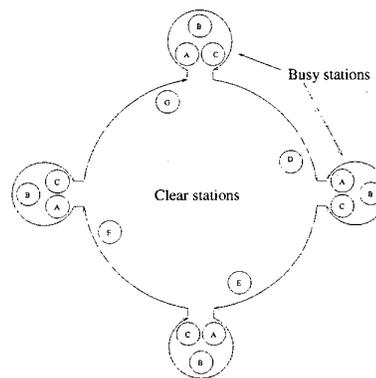


Fig. 1. Illustration of the ERR scheme

episode.

The DQ algorithm allows the network to decide which active sessions should receive degraded service, in terms of loss, delay or both. This scheme uses two logical queues, the  $\alpha$ -queue and the  $\beta$ -queue (see Figure 2). The  $\alpha$ -queue is a short First-In-First-Out (FIFO) queue. Its length corresponds to the longest delay that can be considered “good-service”. If congestion is detected in the  $\alpha$ -queue by its length passing a congestion onset threshold  $T_C$ , a decision is made to redirect one of the active streams of traffic into the  $\beta$ -queue. If congestion continues and subsequent thresholds are crossed, further streams are redirected to the  $\beta$ -queue. When the length of the  $\alpha$ -queue drops below a threshold,  $T_A$ , a selected packet is moved from the  $\beta$ -queue to the  $\alpha$ -queue. Packets from a stream cease being redirected to the  $\beta$ -queue when there are no packets from that session left in the  $\beta$ -queue. The decisions of which stream to redirect to the  $\beta$ -queue at  $T_C$  and which packets to move to the  $\alpha$ -queue at  $T_A$  are chosen so as to fulfill the QoS objectives of the service providers. For example, streams which have been redirected recently could be inhibited from being redirected again, to promote fairness. A timeout threshold  $t_e$  is set for packets in the  $\beta$ -queue.

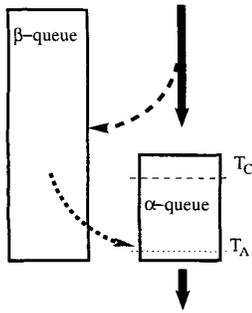


Fig. 2. Illustration of the centralised DQ packet scheduler [10]

It is assumed that packets delayed by less than  $t_G$ , the good service threshold, are all equally useful to the receiver, while packets delayed by more than  $t_G$  are useless. This may be the case when the receiver uses a jitter buffer of length  $t_G$ . Note that packets delayed by more than  $t_e$  are discarded by the network, whereas packets delayed by less than  $t_e$  but by more than  $t_G$  are delivered by the network, but discarded by the application. This allows applications with different delay requirements up to  $t_e$  to use the same network.

#### IV. THE WIRELESS DUAL QUEUE (WDQ)

Although it is effective in wired networks, the original DQ is unsuited to the decentralized wireless uplink. In particular, the problem of polling empty queues is greatly increased. The Dual Queue strategy explicitly gives higher priority to the lightly loaded stations and thus increases the probability of polling a station with no data to transmit. This section describes an efficient method for implementing the Dual Queue on polling networks, based on the ERR algorithm of Section II.

In a centralised queue, congestion is detected by observing the level of the  $\alpha$ -queue. This is impossible in a decentralised system. There are many ways to detect congestion, but the ERR scheme provides an intuitive method. As the network becomes congested the size of the “busy” list will gradually grow up. This increases the polling latency, which is the time taken to poll a station since the scheduler polled the same station last time, for clear stations. If the polling latency exceeds a threshold, the network can be deemed to be congested. This threshold should clearly depend on the maximum delay compatible with “good service”, such as a set fraction  $\theta_C$  of  $t_G$ .

When the load is low and queuing delay at individual nodes can be ignored and the polling latency is the primary source of delay. However, when the network becomes more congested, the polling latency does not reflect the actual packet delays. A potential drawback with this approach therefore is that the network may not be considered congested until the actual packet delays are substantially above the value  $t_G$ .

When congestion is detected, a station should be redirected to the  $\beta$ -queue. The centralised DQ may consider the number of packets from each station in the  $\alpha$ -queue at the time the

threshold is reached, which is impossible in a distributed environment. In our implementation, the station which has transmitted the most data in the previous measurement interval,  $T_M$ , is selected. This is based on the premise that sources are bursty, and a source which has been busy in the past will remain busy for some time to come. Selecting these stations minimises the number of stations which must be redirected in order to effect a given reduction in network load.

In order to enable  $\beta$ -queue stations to return to the  $\alpha$ -queue, there must be a mechanism to clear the backlog while the  $\alpha$ -queue load is light. In the centralised DQ, one of the stations in the  $\beta$ -queue is served whenever the occupancy of the  $\alpha$ -queue is no more than a threshold,  $T_A$ , typically 0. Under ERR, this corresponds to the number of “busy” stations not exceeding  $T_A$ . When that occurs, one station from the  $\beta$ -queue can be served at the end of the round-robin service of the busy stations between the polls of “clear” stations. If  $T_A = 0$ , the station from the  $\beta$ -queue will simply be served once between every pair of polls of the “clear” stations.

Note that for detecting congestion, the natural analog of the  $\alpha$ -queue is the entire set of stations, while for servicing stations in the  $\beta$ -queue, the natural analog is only the busy stations. Because it separates these two, wireless DQ based on ERR can be expected to outperform wireless DQ based on the simple RR strategy.

#### V. PERFORMANCE EVALUATION

In order to evaluate the performance of ERR and WDQ, they were simulated and compared with RR. The simulation set up consists of a single cell infrastructure WLAN [16]. The polling mechanism is essentially that defined in the IEEE 802.11 standard [12] for the “contention free” mode. Although the standard requires that this mode be punctuated by periods of contention based operation, this was omitted in our simulations.

There are ten stations continuously transmitting variable bit rate (VBR) video on the uplink. The downlink is not modeled. The video sources connected with the scheduler generate frames periodically in every  $t_f = 40$  ms. The lengths of the video frames were taken from the real MPEG traces of several standard video sequences [17]. As the sizes of MPEG frames are different, the resulting source is VBR.

The video frames can be very large compared to the maximum length of the MAC Protocol Data Unit (MPDU) defined in the standard. These frames are segmented into packets of size less than or equal to 2312 bytes which is the fragmentation threshold defined in the 802.11 standard. As shown in Table I the time taken to poll an idle station is very much less than the time taken to transmit a typical length packet including the polling overhead.

Simulations were performed for networks with the same load, and raw channel bit rates of 11, 9 and 7.5 Mbps, resulting in system loads of 41%, 50% and 60%. The MAC header and inter-frame spacing were set according to the IEEE 802.11 standard [12]. In order to detect the network congestion in the

TABLE I  
PARAMETERS USED IN SIMULATIONS.

Parameter	value
Good service threshold, $t_G$	80 ms
Measurement interval, $T_M$	20 ms
Congestion cycle time threshold, $\theta_C$	50%
Packet timeout, $t_e$	500 ms
Time to poll an idle station at 10 Mbps channel rate	0.456 ms
Time to transmit a typical length packet (2200 bytes) at 10 Mbps channel rate	2.11 ms

case of WDQ scheduling, the scheduler measures the polling cycle time and compares with some fraction  $\theta_C$  of  $t_G$ . We selected the simplest method to redirect stations from  $\alpha$ -queue to  $\beta$ -queue in WDQ scheduling.

Other system parameters are given in Table I.

### A. Simulation results

Figures 3 and 4 show the times during which users received poor service for medium and high loads considered, under each of the polling strategies. We observed qualitatively the same performance for the low load case. We used  $\theta_C = 0.75$  for the WDQ results presented in these figures.

These graphs are of the form introduced in [10] where a black line is placed for each second in which a station receives "bad service". This is called a "degraded second". Bad service is defined to be either a packet loss or a packet received after an end-to-end delay exceeding threshold  $t_G$ . The assumption is that a user will not easily distinguish between two short periods of degraded service, and so the number of packets involved is not a major concern. With this we attempt to present the user perceivable service quality over the entire video session.

These figures show that ERR polling significantly reduces the total number of degraded seconds experienced by all users compared to simple RR. However the ERR scheme distributes the degradation across all the sessions equally irrespective of the session load. This can be clearly seen in Figures 3(b) and 4(b). During congestion periods the polling latency experienced by the head packet in clear stations increases. If a clear station is backlogged then the scheduler moves that station to the "busy" list and hence it starts to receive service faster. Therefore the packets that have been delivered with small delays under RR experience higher delays under ERR while the packets with larger delays under RR experience less delays under ERR. This resembles the FIFO service discipline, as the packets of all sessions experience the same delay, on average, passing through the FIFO. This makes all users dissatisfied at moderate to high loads.

We define a "degraded episode" as a maximal set of consecutive degraded seconds. The numbers shown in the right hand side of Figures 3 to 4 indicate the number of degraded episodes for each session while the number inside the box is the aggregate number of degraded episodes for all sources. Note that

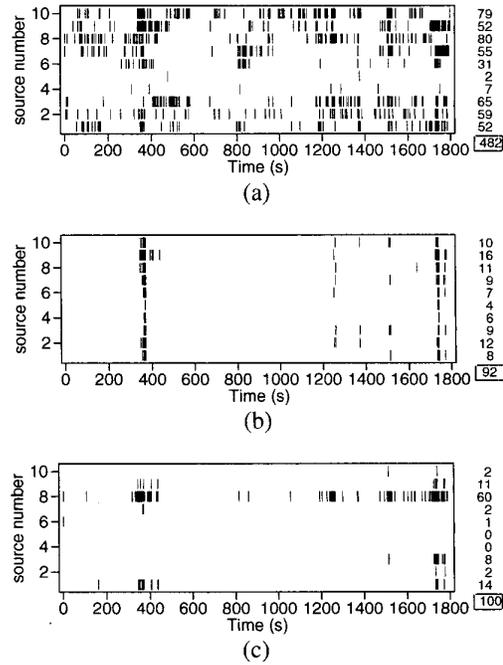


Fig. 3. Bad service times for 50% load using (a) simple RR (b) ERR (c) WDQ.

several degraded episodes may merge to form a single dark line on the graphs. These measures reflect different aspects of the user's perception of the QoS.

In contrast to ERR, WDQ does not only reduce the number of degraded seconds, but also ensures that some of the users receive acceptable service continuously even during periods of high load (see Figures 3(c) and 4(c)). However, the aggregate measurements taken for WDQ are slightly worse than for ERR for lower load. That is because WDQ aims to give users long uninterrupted periods of good service, and is willing to provide very bad service to a subset of users in order to achieve this. Nevertheless, all the users (i.e. including user 8) are better off under WDQ when compared with RR.

Table II shows results for the number of packets which expired during the simulation. In this case, the superiority of ERR over RR is even more marked, with an 88% reduction in expired packets at medium load, and a 100% reduction at low load over the simulation period. The increase in the performance difference for lower loads indicates that ERR is working well to eliminate unnecessary expiration of packets. At higher loads, there will be periods of congestion in which loss is unavoidable, however optimal the polling sequence is.

Once again, WDQ lies between RR and ERR in this statistic. It is worse than ERR because it deliberately delays some packets for very long periods to make way for packets which are less likely to be degraded. This indicates that the current implementation of WDQ would not perform well for data traffic, in which low loss is critical. However, more sophisticated criteria for determining when to relegate stations to the  $\beta$ -queue

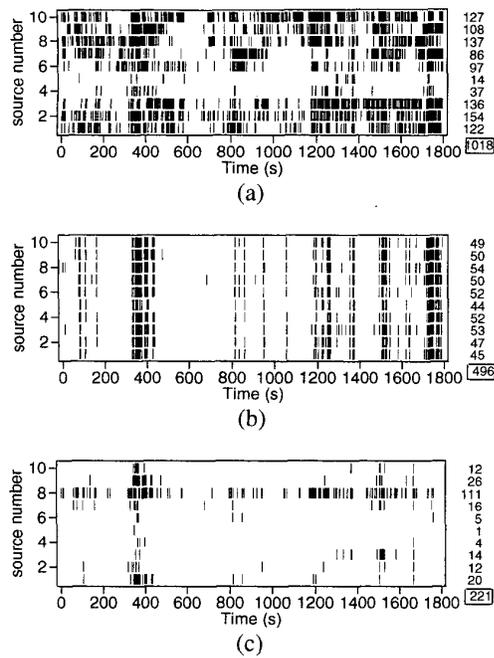


Fig. 4. Bad service times for 60% load using (a) simple RR (b) ERR (c) WDQ.

TABLE II

PERCENTAGE OF PACKETS EXPIRED FOR EACH LOAD.

	41%	50%	60%
RR	0.154	0.462	1.409
ERR	0	0.055	0.449
WDQ	0.012	0.143	0.869

may improve its performance. This is the subject of further investigation.

### B. Selection of $\theta_C$ for WDQ

We specify a certain threshold to detect the network congestion as a set fraction  $\theta_C$  of  $t_G$ . Intuitively, if  $\theta_C$  is small then the scheduler is too aggressive in redirecting the sessions into the  $\beta$ -queue resulting in continuously bad service for certain stations. On the other hand if  $\theta_C$  is large then the scheduler may detect the congestion early enough to start redirection. Therefore more sessions will experience bad performance during congestion episodes simultaneously. Since we implemented the DQ extension on the ERR scheduler, the WDQ behaviour approaches to the ERR behaviour as the value of  $\theta_C$  increases. This can be clearly observed in Figures 5(a), 4(c) and 5(b) which correspond to  $\theta_C = 0.5$ ,  $\theta_C = 0.75$  and  $\theta_C = 1.0$ .

The five cases investigated in these experiments we found that  $\theta_C$  can take range of values in the interval  $[0.675, 0.875]$ . We selected the  $\theta_C = 0.75$  for the results described in Sections V-A, V-C and V-D.

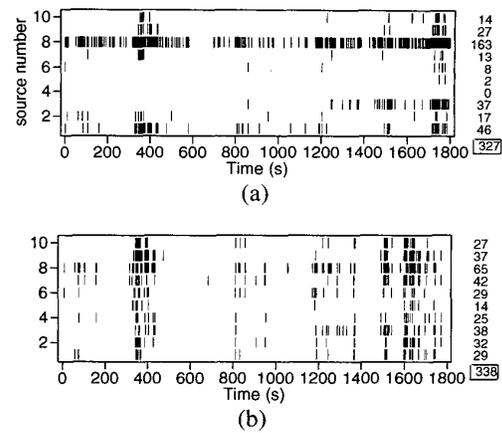


Fig. 5. Bad service times for 60% load under WDQ when (a)  $\theta_C = 0.5$  (b)  $\theta_C = 1.0$

### C. Delay characteristics

The system wide cumulative distribution function (CDF) of queuing delays for the case of 60% load shown in Figure 6 shows that both ERR and WDQ outperform RR scheme. For the lightly loaded sources (i.e. 5 and 6), the proportion of packets experiencing very low delay ( $< 25$  ms) was higher under RR than WDQ. However the rationale behind the WDQ is that if the delay is less than  $t_G$ , reducing the delay further provides no apparent benefit to the user.

Note that there is a sharp rise in the CDF for RR just below  $t_e$ . To understand this, remember that it is during the congestion periods that packets are discarded mostly. Packet delays  $d < 460$  ms ( $= t_e - t_f$ ) corresponds to head-of-queue delay exactly  $d$  ms. Whereas the packet delays  $d = (460 + \delta)$  ms corresponds to head-of-queue delay  $(d + n \cdot 40)$  ms where  $n = 0, 1, 2, \dots$  and  $\delta \in [0, 40)$ . This essentially takes the tail of the distribution beyond 500 ms and stacks 40 ms intervals on top of each other on the interval  $[460, 500)$ . This clearly gives the discontinuous gradient at 460 ms in Figure 7. The reason for the convexity leading up to 460 ms is unclear.

### D. Goodput and throughput characteristics

In these experiments we are interested in delay-oriented throughput, i.e. the amount of traffic passing through to the destination within delay  $t_G$ . Consider two statistics related to throughput.

- Goodput: packets delivered before  $t_G$  as a percentage of offered load
- Throughput: packets delivered as a percentage of capacity

Figure 6 shows that, for an offered load of 60% there is a negligible loss for all three disciplines. Therefore the throughput is roughly 60%. For RR the goodput is 88.4% of the offered traffic or 53% of the capacity. For ERR the goodput is 96.2% of the offered traffic or 57.7% of the capacity and for WDQ it is 97.2% of the offered traffic or 58.3% of the capacity. Note that

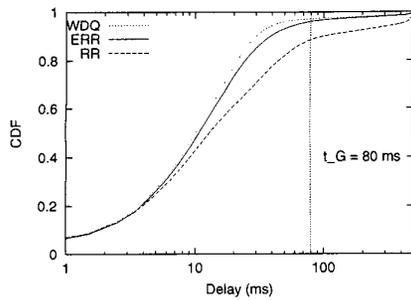


Fig. 6. System wide CDF of queuing delays at 60% load.

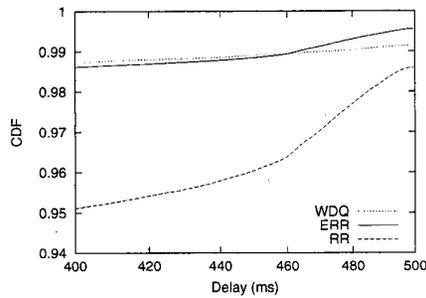


Fig. 7. Magnified view of top right hand corner of Figure 6.

even though the goodput achieved by both ERR and WDQ are the order of same magnitude, WDQ schedules the packets so as to increase the number of satisfied users as illustrated in Figure 4. As the offered load increases the throughput increases while the goodput decreases.

## VI. CONCLUSIONS

This paper has presented two novel polling strategies, namely Embedded Round Robin (ERR) and Wireless Dual Queue (WDQ), suitable for a wireless uplink. Importantly, neither of these schemes requires state feedback from the remote stations except a single bit to indicate that a station has more packets awaiting service. This single bit feedback is, however a common feature of existing protocols.

The basic idea behind ERR is to reduce the polling overhead caused by polling stations with no data to transmit. The saved bandwidth is then allocated to "busy" users, thereby increasing the overall network throughput. This work has clearly indicated the advantages of polling stations depending on the availability of data at the queues. Simple schemes with minimum overhead for predicting and classifying stations as "clear" or "busy" in the case of heterogeneous traffic is part of ongoing research. The ERR scheme distributes the degradation across all the sessions equally, irrespective of the session load. This is the major drawback of ERR when we consider real time sessions, as all the sessions experience bad perception during congestion episodes.

WDQ overcomes this problem by adapting the wireline dual

queue algorithm to the wireless uplink. The WDQ effectively eliminates the major drawback of the ERR scheme, allowing the low bandwidth users to continue to receive good service even during congestion episodes. We have presented a simple mechanism to detect network congestion and to identify stations which cause network congestion centrally at the base station without explicit overhead. In this study, the WDQ discipline was integrated with the ERR discipline. Simulation results show that the WDQ is good for real time services where long periods of low delay are important. However there is a cost associated with this achievement, for example it causes more packets to time out than the ERR scheme.

Both ERR and WDQ have complementary advantages and both outperform standard RR as measured by the performance metrics studied.

## REFERENCES

- [1] G. Anastasi, L. Lenzini, E. Mingozzi, "MAC protocols for wideband wireless access: evolution towards wireless ATM," *IEEE Personal Comm.*, Oct. 1998, pp. 53-64.
- [2] D. Ferrari, D. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE J. Sel. Areas Comm.*, April 1990, vol. 8, no. 4, pp. 368-79.
- [3] N.R. Figueira, J. Pasquale, "Leave-in-time: a new service discipline for control of real-time communications in a packet-switching network," *Proc. ACM SIGCOMM'95*, August 1995, pp. 207-18.
- [4] D. Verma, H. Zhang, D. Ferrari, "Delay jitter control for real-time communication in a packet switching network," *Proc. IEEE TriCom '91*, April 1991, pp. 35-43.
- [5] A. Demers, S. Keshav, S. Shenker, "Analysis and simulation of a fair queuing algorithm," *Proc. SIGCOMM 89*, Sept. 1989, vol. 19, no. 4, pp. 1-12.
- [6] A.K. Parekh, R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Trans. Networking*, June 1993, vol. 1, no. 3, pp. 344-57.
- [7] S.J. Golestani, "A self-clocked fair queuing scheme for broadband applications," *Proc. IEEE INFOCOM '94*, 1994, pp. 636-46.
- [8] D. Saha, S. Mukherjee, S.K. Tripathi, "Carry-over round robin: a simple cell scheduling mechanism for ATM networks," *IEEE/ACM Trans. Networking*, Dec. 1998, vol. 6, no. 6, pp. 779-96.
- [9] M. Shreedhar, G. Varghese, "Efficient fair queuing using deficit round robin," *IEEE/ACM Trans. Networking*, 1996, vol. 4, no. 3, pp. 375-85.
- [10] D.A. Hayes, M. Rumsewicz, and L.L.H. Andrew, "Quality of service driven packet scheduling disciplines for real-time applications: looking beyond fairness," *Proc. IEEE INFOCOM '99*, 1999, vol. 1, pp. 405-12.
- [11] J. Nagle, "On packet switches with infinite storage," *IEEE Trans. Comm.*, Apr. 1987, vol. COM-35, no. 4, pp. 435-38.
- [12] P802.11, Wireless medium access control (MAC) and physical layer (PHY) working group, IEEE 802.11 draft standard, "Wireless medium access control (MAC) and physical layer (PHY) specifications," IEEE Stds. Dept., July 1996, D5.0
- [13] N. Movahhedinia, G. Stamatelos, H.M. Hafez, "Polling-based multiple access for indoor broadband wireless systems," *Proc. PIMRC 95*, 1995, vol. 3, pp. 1052-6.
- [14] R. Kautz, A. Leon-Garcia, "A Distributed self-clocked queuing architecture for wireless ATM networks," *Proc. PIMRC'97*, 1997, vol. 1.1, pp. 189-93.
- [15] C.-S. Chang, K.-C. Chen, "Guaranteed quality-of-service wireless medium access by packet-by-packet generalized processor sharing algorithm," *Proc. ICC'98*, June 1998, vol. 1, pp. 493-7.
- [16] R.S. Ranasinghe, D. Everitt, L.L.H. Andrew, "Fair Queuing scheduler for IEEE 802.11 based wireless multimedia networks," *Multimedia, Mobility and Teletraffic in Wireless Communications (MMT'99)*, Oct. 1999, vol. 4, pp. 241-51.
- [17] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," *Proc. 20th Annual Conf. Local Comp. Network*, Minneapolis, CA, Oct. 1995, pp. 397-406.