Implementation of the Multiclass Priority Fair Queuing (MPFQ) Algorithm for Extending Quality of Service in Existing Backbones to Wireless Endpoints

Jay R. Moorman
jrmoorma@uiuc.edu

John W. Lockwood
lockwood@ipoint.vlsi.uiuc.edu

University of Illinois at Urbana-Champaign

Abstract

This paper presents the implementation of a wireless scheduling algorithm that provides backbone Quality of Service (QoS) support to mobile stations. The algorithm described in this paper maps all existing ATM traffic types into priorities and weights for a Multiclass Priority Fair Queue (MPFQ) scheduler.

The MPFQ scheduler is designed to operate in a wireless base station. A prototype driver has been developed for the Netwave AirSurfer Plus Wireless PCMCIA card. Simulation results are included that demonstrate the advantages in reduced delay for real-time traffic flows when using the MPFQ scheduler instead of a fair queue scheduler.

1 Introduction

The notion of QoS on a wireless link has recently received substantial attention. Work has been done to support certain bandwidth and delay requirements over bandwidth limited links [1, 2, 3]. Some of the work has been specific to Asynchronous Transfer Mode (ATM) traffic [1, 4, 5], focusing on call admission, bandwidth allocation, or scheduling for a particular type of traffic.

Variations to fair queuing have been developed that address issues specific to a wireless link [6, 7, 8]. In [6] an algorithm was proposed that modifies a fair queuing algorithm for Wireless Packet Scheduling (WPS). This algorithm compensates for wireless channel errors that tend to be both bursty and location-dependent.

This paper is a companion paper to [9] for the scheduling of ATM traffic classes over a wireless link. In this paper, implementation details of the scheduling algorithm and the implementation of our Linux-based wireless testbed are presented. Simulation results are presented for comparison of the MPFQ algorithm with a standard fair queuing algorithm.

The Multiclass Priority Fair Queuing (MPFQ) algorithm transforms ATM traffic class into the parameters for the wireless channel packet transmission. The scheduler uses a combination of WPS, a Weighted Round Robin (WRR) scheduler with wireless compensation, and a simple FIFO. By calculating the bounds on the delay and reserving bandwidth, each connection's QoS guarantees, within the schedulable region, can be met in the worst case and exceeded in general.

2 Wireless Access Model

In this paper, we assume a high-speed wired ATM backbone that is extended to service mobile stations in a packetized wireless cellular network. The wireless access network is divided into partially overlapping cells that transmit on different logical channels. Each cell contains a base station that is connected to the wired network. The base station schedules packets for transmission on the wireless media. The channel is said to be in error for a particular flow if either the sender or the receiver of the flow detects a local error. All errors are assumed to be location dependent and typically bursty. QoS is supported by meeting parameters that are negotiated at call admission. These parameters, including cell rate and maximum delay, are then guaranteed by the scheduler for the lifetime of the connection.

Notation	Definition
$P_{i,n}$	Packet n of flow i.
ϕ_{k_i}	The weight for flow i in class k.
λ_{k_i}	The arrival rate of flow i in class k.
$A_{i,n}$	Arrival time of packet n in flow i.
$s_{i,n}$	The virtual start time for packet n in flow i.
$f_{i,n}$	The virtual finish time for packet n in flow i.
L_p	The length of fixed packets (cells) in all flows.
C	Channel capacity.
C_k	The relative capacity of class k.
F_k	The set of all flows in class k.
D_k^{max}	The maximum delay for class k HOL packet.
Q_{k_i}	The buffered queue length of flow i in class k.
B	Bounded amount of system compensation.
T_b	The time period for added buffering.
T_w	The timeout period for errored channels.

Table 1: MPFQ Notation

In our testbed, a basestation is implemented running the linux operating system using a PC. The PC has both an ATM interface and a wireless LAN interface. The MPFQ algorithm running on this machine schedules the transmission of the packets to the mobile stations.

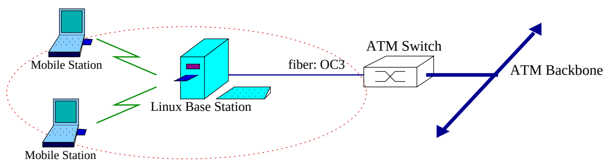

Figure 1: Scheduler Testbed

3 Multiclass Priority Fair Queuing (MPFQ)

The MPFQ scheduler is based on a multiclass priority algorithm that separates flows according to each general class of traffic. These classes as specified in [10] for backbone networks include CBR, rt-VBR, nrt-VBR, ABR, and UBR. The scheduler adheres to strict priorities between levels which is both desirable and necessary as shown in [9]. Within each level, a specific scheduling method is used depending on the necessary traffic requirements. By extending the backbone QoS parameters onto the wireless communication link, the end-to-end QoS requirements can be satisfied and fairness can be maintained for delay sensitive traffic.

The MPFQ algorithm defines the mapping of the traffic classes into priority levels and weights at call admission. Real-time traffic uses a fair queuing algorithm specifically adapted to the wireless medium. The non-real-time traffic uses a WRR scheduler with an adaptation for wireless compensation. The remaining traffic receives best effort service.

4 MPFQ Priority Mapping

The mechanism by which the MPFQ separates ATM QoS levels into priorities along with their associated parameters is shown in Table 2. One logical server implements scheduling policies at each level. Conceptually, this can be viewed as a different server at each priority level as shown in Figure 2.

Priority	Traffic Type	Parameters	Queuing Policy
1	CBR	PCR, CTD	WPS
2	rt-VBR	SCR, MBS, CTD	WPS
3	nrt-VBR & ABR-MCR	SCR, MBS & MCR	WRR Scheduler w/ Compensation
4	ABR	Unspecified	Recirculating FIFO
5	UBR	Unspecified	FIFO

Table 2: ATM Prioritized Traffic

The highest two priorities of the scheduler are for real time data and require tight bounds on the guaranteed delay. These delays are satisfied and the channel is allocated fairly through the use of the wireless packet scheduling model. These levels take priority over the non-real-time traffic and have been divided into CBR (level 1) and rt-VBR (level 2) classes for better CBR performance. To enable different discard policies, but not force a flow to relinquish its compensation for unsent packets, each flow has a separate packet queue and slot queue. The slot queues only contain the start/finish time data for packets in the flow and a pointer to the HOL packet in the packet queue.

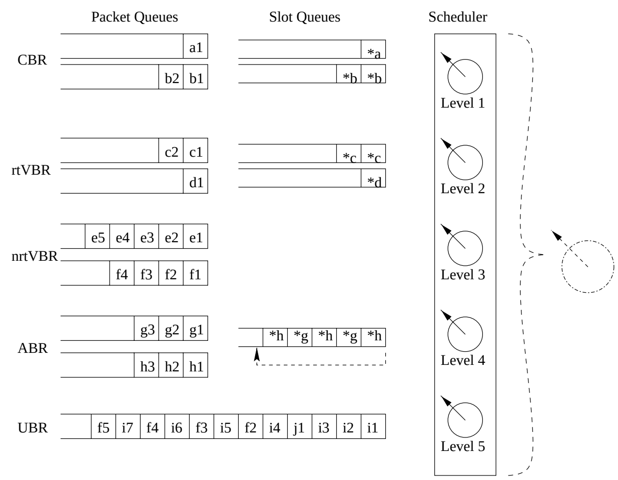

Figure 2: MPFQ Base Station Scheduler

The third priority (level 3) in the system is used for flows that require a guaranteed bandwidth without loss, but are not constrained by tight delays. These flows do not require the extensive fair queue scheduling, instead MPFQ uses a simple WRR scheduler to ensure that traffic bounds are met over time. Compensation for wireless error is taken into account by using credits and debts as in [6]. In effect, when a flow cannot transmit, it relinquishes its slot to another flow that can transmit. This mechanism allows the scheduler to credit flows to reachable hosts and borrow bandwidth from flows to unreachable hosts.

ABR traffic is given strictly higher priority than UBR connections and is further divided across two different priorities (Level 3 and Level 4). This enables the MPFQ to meet the Minimum Cell Rate (MCR) guarantee for ABR traffic. The split priority levels allow part of the ABR traffic (ABR_{MCR}) to obtain guaranteed access to the channel at the same priority as the nrt-VBR class. The traffic that arrives at rates above the MCR may use the remaining bandwidth only after all higher-priority traffic has been serviced. Since the ABR traffic is serviced across priorities it must remain in per flow packet queues. However, for the level 4 priority traffic there is a single slot queue which acts as a FIFO for the excess ABR traffic. Each packet that arrives gets placed in the flows packet queue and a slot is reserved in the level 4 slot queue. If the scheduler sees an error for the HOL flow the slot is recirculated to maintain bandwidth allocation without blocking transmission.

The lowest priority level uses a simple FIFO for the scheduling. UBR traffic is placed in the FIFO consisting of only a single level 5 packet queue. Since no guarantees are made on delay or bandwidth the scheduler can drop packets as needed. This dropping can occur either for outgoing data due to channel error, or for incoming data due to a full buffer.

4.1 MPFQ Weight Mapping

To work within the context of weighted fair queuing, the parameters of QoS need to be mapped into individual weights. The algorithm that maps traffic classes into weights generates three parameters for each traffic connection. Bandwidth is reserved in order to satisfy the cell rates; the weights are assigned for the fair queuing algorithm in order to satisfy delay; and buffers are allocated in order to guarantee a flow will not lose data.

In general, the traffic rate is used as a percentage of bandwidth. The call admission delay parameter is used to find the weight since it is inversely proportional to the desired weight. The weights for ABR traffic are set by the current flow rate in the most recent RM cell [11]. The necessary buffer space is reserved at each level. These per-flow reserved buffer allocations provide sufficient memory to avoid loss while higher priority flows drain their queues. Additional buffer space is added for wireless compensation. This amount is the lesser of $PCR * T_b$ or $(SCR * T_b) + MBS$. The weight and reserved buffer space parameter mappings as derived in [9] are shown in Table 3.

4.2 MPFQ Delay Bounds

A vital part of the MPFQ algorithm is the analysis of the delay bounds for the traffic flows at each priority level in the hierarchy. These bounds, determined in [9], can be used along with the individual connection weights both for call admission and for scheduling.

After bounding delay for the error-free model, the effect of wireless channel error must be considered. Any flow that has been backlogged and unable to send due to an errored channel should receive compensation. This can be accomplished by a number of methods including complete access to the channel, linear access, weighted access, or exponential access.

The MPFQ method captures the remaining bandwidth once a channel becomes error-free. In order to bound the time that the flow may be delayed, the algorithm limits compensation. The total amount of wireless compensation is bounded by B bits, which in effect increases the worst case bounded delay by B/C.

For CBR traffic the bounded wireless delay is:

$$D^{max}_{wireless_{CBR}} = \frac{L_p}{C} \cdot \left[1 + \frac{\sum_{j \in F_{CBR}} \phi_{CBR_j}}{\phi_{CBR_i}} \right] + \frac{B}{C} \quad (1)$$

For rt-VBR traffic the bounded wireless delay is:

$$D^{max}_{wireless_{rtVBR}} = \frac{B}{C} + \frac{L_p}{C_{rtVBR}} \cdot \left(1 + N_{CBR} + \frac{\sum_{j \in F_{rtVBR}} \phi_{rtVBR_j}}{\phi_{rtVBR_i}} \right) \quad (2)$$

where the effective reduced rate is:

$$C_{rtVBR} = C - \sum_{j \in F_{CBR}} \lambda_{CBR_j} \quad (3)$$

5 MPFQ Scheduler

The MPFQ scheduler calculates a virtual start and finish time for each real-time packet. When a flow is in error, the queue is not drained and must later be compensated. Once the channel becomes clean the flow can send at an increased rate since it has packets with low virtual times. Non-real-time traffic is compensated through the use of credits/debts for flows ahead/behind their error-free schedules. However, the amount of compensation is bounded by the available buffering reserved at call admission. If the buffer space becomes full, the negotiated discard policy must then be initiated. The discard policy is also invoked when the delay on a packet has been exceeded. When a flow is forced to discard, it will release a packet according to the discard policy but will maintain the slot for proper wireless compensation and long term fairness.

To decide which packet to send next, the scheduler finds a HOL packet from the highest priority server with an error-free channel. Real-time traffic uses the minimum finish time while non-real-time traffic uses a WRR schedule. The scheduler tracks errored channels by maintaining a boolean flag for each flow. This information is provided by the MAC layer using a one-step prediction as described in [6, 9]. If the scheduler does not receive a CTS from the mobile, the channel is assumed to be in error. After a timeout period T_w the channel is probed again to determine a new status.

5.1 MPFQ Algorithm

The MPFQ algorithm consist of two major routines to account for incoming and outgoing packets. The *Receive_Packet* routine accepts packets into the system. This procedure is passed a packet along with the traffic class and flow number of the packet. The *Schedule_Packet* routine determines what packet is to be transmitted onto the wireless channel. This procedure returns with the packet to send.

```
Receive_Packet(packet p, class k, flow i)
    If (k == Real-Time Traffic)
        If (Q_{k_i} < Q_{k_imax})
            s_{i,p} = max(V(A_{i,p}), f_{i,p-1});
            f_{i,p} = s_{i,p} + L_p/φ_i;
            Insert(pq[i], p);
            Insert(sq[i], s_{i,p}, f_{i,p});
        Else Invoke Discard Policy;
    ElseIf (k == Non-Real-Time Traffic)
        If (Q_{k_i} < Q_{k_imax})
            Insert(pq[i], p);
            If (Type(i) == ABR)
                Enqueue(sq[ABR], p);
        Else Invoke Discard Policy;
    ElseIf (k == Best Effort Traffic)
        Enqueue(pq[UBR], p);
```

When a packet is received, the algorithm reacts according to the packet's type. If the traffic is real-time

Traffic	Rate	Weight (ϕ_n)	Buffer Size without Wireless Compensation
CBR	PCR	$\frac{1}{maxCTD}$	1
rt-VBR	SCR	$\frac{1}{maxCTD}$	$\left(MBS - \frac{MBS}{PCR} \cdot SCR\right) + \frac{L_p * N_{CBR}}{C} \cdot SCR$
nrt-VBR	SCR	N/A	$\left(MBS - \frac{MBS}{PCR} \cdot SCR\right) + \left(\frac{L_p * N_{CBR}}{C} + \frac{\sum Q_{rtVBR}}{\sum SCR_{rtVBR}}\right) \cdot SCR$
ABR	MCR	N/A	$\left(\frac{L_p * N_{CBR}}{C} + \frac{\sum Q_{rtVBR}}{\sum SCR_{rtVBR}}\right) \cdot MCR$
UBR	N/A	N/A	N/A

Table 3: ATM Traffic Mappings

and buffer space is available, the packet is inserted into the packet queue. The start/finish times are calculated and a slot reserved in the slot queue. If the packet is non-real-time, the packet is enqueued for transmission. For ABR a slot in the slot queue is also reserved. If the traffic is UBR it is added to the FIFO. If no buffer space is available for any type of traffic the appropriate discard policy is invoked. The code for scheduling packets is shown below:

```
packet = Schedule_Packet()
    If (Level1 Packets ≠ ∅) /* (CBR) */
        for all (j ∈ F1)
            i = min(f_{i,p});
            If (flow(i) has clean channel)
                Dequeue(sq[i]);
                p = Dequeue(pq[i]);
                Return(p);
    ElseIf (Level2 Packets ≠ ∅) /* (rtVBR) */
        for all (j ∈ F2)
            i = min(f_{i,p});
            If (flow(i) has clean channel)
                Dequeue(sq[i]);
                p = Dequeue(pq[i]);
                Return(p);
    ElseIf (Level3 Packets ≠ ∅) /* (nrtVBR) */
        for all (j ∈ F3)
            i = next flow in WRR scheduler
            If (credit(i) > 0) /* Swap Credit/Debit Flows */
                Credit(i) − −;
                i = Max_Debt(Level3 Clean Flows);
                Debt(i) − −;
            If (flow(i) has dirty channel)
                Debt(i) + +;
                i = Next Clean Level3 Flow
                Credit(i) + +;
            If (Type(i) = nrtVBR)
                p = Dequeue(pq[i]);
                Return(p);
            ElseIf (Type(i) = ABR && Rate < MCR_i)
                Remove(sq[ABR], i);
                p = Dequeue(pq[i]);
                Return(p);
    ElseIf (Level4 Packets ≠ ∅) /* (ABR) */
        for all (j ∈ F4)
            If (flow(i) has clean channel)
                Dequeue(sq[ABR]);
                p = Dequeue(pq[i]);
                Return(p);
    ElseIf (Level5 Packets ≠ ∅) /* (UBR) */
        for all (j ∈ F5)
            i = next flow in FIFO
            If(flow(i) has clean channel)
                p = Dequeue(pq[UBR]);
                Return(p);
```

To schedule the next packet for the wireless channel, the MPFQ scheduler traverses down the priority levels until a nonempty level is found. At level 1 or level 2, the packet with the minimum finish time in a non-errored flow is chosen. Level 3 will choose the next cell in the precomputed WRR schedule. If this flow has credit, it is swapped with a flow that has debt and a clean channel. If no swapping is needed and the flow has a clean channel, it is sent. In the case of an errored channel another clean flow is chosen to send in this packets place. In all situations the credit/debt of each respective flow is properly updated. This level also checks if the packet is from an ABR flow, and thus if the rate needs to be compared to the MCR. Level 4 sends the first packet from a clean flow in the slot queue FIFO. Slots from errored flows are recirculated to avoid loss and maintain bandwidth. The level 5 scheduler is the first packet from a clean flow out of the packet queue FIFO with no guarantees.

At any time that a packet is sent, the packet is removed from the proper packet queue. The appropriate slot queue is also updated to reflect the removal of the transmitted packet. In the case of a discarded packet, the proper packet is removed from the packet queue but the slot queue will retain the flows slot for accurate wireless compensation.

5.2 Linux Testbed Implementation

The MPFQ scheduler augments the functions of an access point in a wireless LAN. Our prototype access point is implemented using a Linux Base Station running a device driver developed for the Netwave AirSurfer Plus.

This driver operates in conjunction with the PCMCIA services in the Linux kernel. The object file is compiled as a loadable module which is invoked upon insertion of a PCMCIA wireless device. Details of the driver can be found in [12]. The latest version of the driver (v1.0.2) can be downloaded from [13].

6 Simulation Results

Simulations were performed on various flow scenarios to test the MPFQ algorithm. These results were then analyzed to compare the performance of the MPFQ algorithm with a single-class fair queue scheduler. To illustrate the core functionality of the algorithm, three flows including a CBR, a rt-VBR, and a nrt-VBR flow, were generated. The rates were chosen so as to stress the scheduler into a congestive state. The flows in Table 4 are fully representative of a scenario with a multiple number of flows of each class combined into an aggregate rate for that traffic type.

A traffic source was built to generate input for the MPFQ scheduler [9]. The traffic generator produces CBR traffic at a constant rate. VBR traffic is gener-

Flow	CBR	rtVBR	nrtVBR
PCR	0.20	0.95	0.40
SCR	0.20	0.40	0.25
MBS	1	200	400
Level	1	2	3

Table 4: Simulation Traffic Flows

ated using a 2-state Markov process with the first state bursting at the PCR and the second state sending at a reduced rate. The output is shaped to produce a traffic source that conforms to the SCR and does not burst for more than the MBS. During the simulation, 30,000 packets were generated to provide statistically accurate measurements of true worst-case delay. Results are shown below:

Flow	CBR	rtVBR	nrtVBR
MPFQ Worst WC Delay	1	72	601
FQ Worst WC Delay	4	342	620
MPFQ Average WC Delay	1	69.71	525.35
FQ Average WC Delay	3.54	248.82	498.27

Table 5: MPFQ/FQ Simulation Comparison

The worst case traffic scenario was run through the simulator 100 times to track and record the average worst case delay and the absolute worst case delay. This produced the results in Table 5 which show the advantage in bounded delay of the MPFQ algorithm. The real-time traffic delay using the MPFQ scheduler was significantly lower than the single-class FQ scheduler.

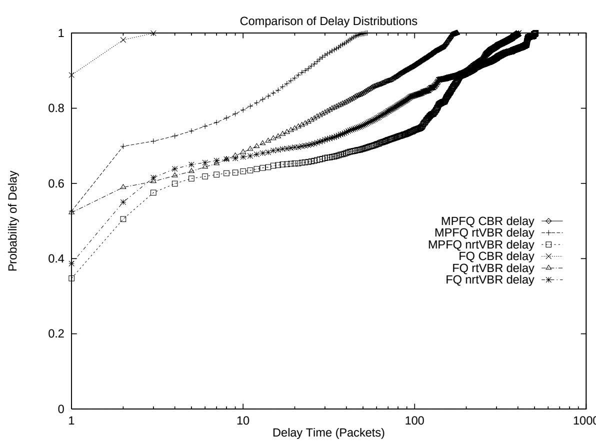

Figure 3: Cumulative Distribution of Delay

To further show the advantage in the prioritized algorithm, Figure 3 shows the distribution of the packet delay times from a single run both for the MPFQ scheduler and the FQ scheduler. The CBR traffic under FQ is delayed longer than that of its MPFQ counterpart. This MPFQ CBR traffic never experiences a delay greater than 1 packet. A significant amount of rtVBR traffic suffers from longer delays in the FQ case. As expected, only the nrtVBR traffic has a consistently smaller delay with fair queueing. The reduced delay in the non-real-time traffic comes at the expense of added delay in real-time traffic.

7 Conclusion

This paper presented the implementation details of a Multiclass Priority Fair Queuing (MPFQ) algorithm. This algorithm, operating in a base station as the wireless packet scheduler, enables mobile stations to seamlessly transmit QoS-enabled traffic flows to a backbone network.

The MPFQ scheduler prioritizes flows of traffic according to the specific class. Bandwidth is shared fairly among all flows in the system. The separation of classes serves to provide minimum delay guarantees to real-time traffic. This improved delay performance is demonstrated by simulation.

References

[1] J. G. Kim and M. Krunz, "Effective bandwidth in wireless ATM networks," in *MOBICOM '98*, (Dallas,Texas), pp. 233–241, Oct. 1998.

[2] G. R. J. Linnenbank, P. Venkataram, P. J. M. Havinga, S. J. Mullender, and G. J. M. Smit, "A request-TDMA multiple-access scheme for wireless multimedia networks," in *MoMuC-3*, (Princeton,New Jersey), Sept. 1996.

[3] S. Choi and K. G. Shin, "Uplink CDMA systems with diverse QoS guarantees for heterogeneous traffic," in *MOBICOM '97*, (Budapest,Hungary), pp. 120–130, Sept. 1997.

[4] W. Su, "Bandwidth allocation strategies for wireless ATM networks using predictive reservation," in *GLOBECOM '98*, (Sydney, Australia), Nov. 1998.

[5] A. Acampora and M. Naghishineh, "Control and quality of service provisioning in high-speed microcellular networks," *IEEE Personal Communications*, vol. 1, no. 2, 1994.

[6] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," in *SIGCOMM '97*, (Cannes, France), 1997.

[7] S. Lu, T. Nandagopal, and V. Bharghavan, "A wireless fair service algorithm for packet cellular networks," in *MOBICOM '98*, (Dallas,Texas), pp. 10–20, Oct. 1998.

[8] P. Ramanathan and P. Agrawal, "Adapting packet fair queueing algorithms to wireless networks," in *MOBICOM '98*, (Dallas,Texas), pp. 1–9, Oct. 1998.

[9] J. Moorman, J. Lockwood, and V. Bharghavan, "Multiclass priority fair queuing for support of quality of service over hybrid wired and wireless networks." Submitted to MOBICOM '99.

[10] "Traffic Management Specification v4.0." ATM Forum Document AF-TM-0056.000, Apr. 1996.

[11] M. Bossardt, J. W. Lockwood, S. M. Kang, and S.-Y. Park, "Available bit rate architecture and simulation for an input-buffered and per-vc queued ATM switch," in *GLOBECOM '98*, (Sydney, Australia), Nov. 1998.

[12] J. R. Moorman, "Supporting quality of service on a wireless channel for ATM wireless networks," Master's thesis, University of Illinois, Jan. 1999.

[13] J. R. Moorman, "AirSurfer Plus device driver." Available via WWW at `http://ipoint.vlsi.uiuc.edu/wireless`, 1999.