

# Experimental Validation of Middleware-based QoS Control in 802.11 Wireless Networks (Invited paper)

Wenbo He \*  
wenbohe@cs.uiuc.edu

Hoang Nguyen †  
hnguyen5@uiuc.edu

Klara Nahrstedt  
klara@cs.uiuc.edu

Department of Computer Science  
University of Illinois at Urbana-Champaign  
Champaign, IL, 61801, United States

## Abstract

*Due to the shared medium nature of wireless networks, the uncertainties caused by collisions and interferences make the Quality of Service (QoS) issue harder than its wired counterpart. Many publications have been focused on network and MAC layer design to address the QoS issue in wireless networks. However, the middleware design has been overlooked. For QoS support, we need to map the QoS requirement of applications to performance metrics. Middleware is the place we do such mapping. In this paper, we use packet level priority to bridge the QoS requirements and performance. Through middleware priority adaptation, we aim to make the premium traffic meet the QoS requirement. we study the impact of middleware priority adaptation on QoS performance, including bandwidth and end-to-end delay, via experiments with multimedia flows over IEEE 802.11 environment. Our evaluation is based on experiments in both WLAN and ad hoc network environment. Our investigation shows that middleware adaptation is efficient in assisting to achieve QoS in many scenarios.*

## 1. Introduction

Multimedia services over wireless ad hoc networks or multihop WLAN are becoming pervasive in applications such as music sharing, voice telephony and others. The scenarios where multimedia applications and ordinary data traffic co-exist on wireless ad hoc networks or multihop wireless LAN, include, but are not limited to:

- In a neighborhood community, people play games, share music or movies in a wireless mesh network, while other services such as email, ftp, and www are

being used simultaneously.

- In a university auditorium, the audio or video recording the speakers' speech or activities are sent to the storage through multihop wireless links. Meanwhile, the audience is sending or receiving email, browsing websites using the same multihop WLAN resources.

As the demand of multimedia services over 802.11 wireless networks increases, we are facing the challenge of offering premium applications with Quality of Service (QoS) requirements over wireless networks, since the bandwidth of a wireless link is limited, unpredictable, and the channel capacities and error rates are time-varying in comparison to wired networks. The existing protocols follow several directions to achieve the goal of QoS in wireless networks:

The first type of QoS protocols on wireless networks is based on admission control and conflict resolution. [1] provides an admission control scheme according to the resource prediction by probing method. The uncertainties of wireless network may cause false admissions, since admitting new traffic increases contention in the shared channel due to the admission control. [2] proposes an admission control and dynamic bandwidth management scheme that provides fairness and a soft rate guarantee without using distributed MAC-layer weighted fair scheduling. [3] addresses the contention-aware admission control so that the admitted flows in the network do not exceed network capacity. [4] addresses the QoS in wireless networks by utility-based resource allocation.

The second type of QoS protocols for wireless networks is based on MAC layer scheduling, including TDMA scheduling protocols [5] [6] and IEEE802.11 based protocols [7]. QoS over TDMA based protocols is obtained by assigning different time slots to different nodes within the same contention range. TDMA based MAC protocols re-

quire time synchronization among nodes in each contention area. It is very expensive to provide synchronization in wireless ad hoc networks.

IEEE 802.11 based protocols deal with the contention within a single channel and at any time slots. IEEE 802.11 uses carrier sense multiple access with collision detection (CSMA/CD) and has two modes: PCF (Point Coordination Function) and DCF (Distributed Coordination Function). IEEE 802.11 PCF is a centralized mechanism [8], and provides high packet delivery ratio. IEEE 802.11 PCF supports time bounded delivery for real time flows. However, it requires the support of the base station, and the synchronization is required among wireless nodes. Another drawback of IEEE 802.11 PCF scheme is it has large overhead for unnecessary polling. IEEE 802.11 DCF is a listen-before-transmit scheme. IEEE 802.11 DCF [9], [10], [11], [12] provides service differentiation by assigning different service parameters, such as backoff-interval size, DIFS, and maximum frame length of a packet, to different service classes. However, the IEEE802.11 DCF differentiation service only guarantees service ratio of realtime flows to non-realtime flows, and MAC layer itself is not enough to provide end-to-end QoS.

The third type of QoS protocols uses cross-layer design that has attracted much attention in recent publications [13] [14], since it is becoming increasingly clear that optimizing within layers is insufficient to obtain enough performance gains. Cross-layer solutions can yield significant improvement in performance by tight coupling the layers in wireless systems. Currently, most of the cross-layer design is aimed at lower layers, e.g. physical layer, MAC layer and network layer.

However, the mapping from end-to-end QoS requirement given by application layer to lower-layer network performance is missing in the existing cross-layer designs. Middleware is the right place to do such mapping since middleware layer has direct access to application requirements as well as it can access information about resource availability in wireless network. Another concern for QoS management in our work is compatibility. Since 802.11a/b/g wireless products are widely-adopted in market, it is important not to modify the MAC layer. Therefore, our system will lie from network layer to middleware layer. In this paper, we will consider statistical QoS requirements and their mapping to packet-level priority in middleware. We use bandwidth as the QoS metric for WLANs, and end-to-end delay as the QoS metric for wireless ad hoc networks. Packet-level priority is an important parameter in wireless packet scheduling and strongly influences the overall end-to-end delay of multimedia flows as packets get scheduled along the path. The middleware mapping is a hard problem due to the dynamics of the wireless channel, interference at hosts and little support for QoS control in the existing

802.11 MAC layers. Furthermore, the mapping needs to be robust and relatively stable to achieve acceptable user satisfaction. We have designed the priority mapping and its adaptation based on control theory. We notice that feedback control theory has received great attention recently in quality of service (QoS) on a variety of computer system platforms [15] [16] [17] [18]. The middleware priority adaptation mechanism follows a light-weighted and flexible design. We validate the priority mapping adaptation framework in our 802.11b testbed, using WLAN as well as ad hoc networks. The experimental objective is to estimate how effective the middleware mechanism is within the QoS management. The current results indicate that our priority-mapping middleware framework yields desired QoS levels within acceptable statistical bounds.

This paper is organized as follows. Section 2 describes our methodology and architecture. Section 3 shows experimental setup, and demonstrates the test results. Section 4 provides more discussions and concludes the paper.

## 2 Middleware Design Within Cross-layer Framework

Our middleware design will focus on two wireless scenarios: WLAN and multi-hop ad hoc. Since our middleware will use QoS metrics to control the system, we need to define metric for each scenario.

In WLAN, wireless nodes share the same channel to communicate with base station. In this scenario, end hosts connect with the base station within one hop. An application's QoS concern is how much bandwidth it can get. Therefore in WLAN scenarios, we use bandwidth as the QoS metric, and want to adjust packet-level priority to meet bandwidth requirement of the application. In ad hoc networks, an application transmits packets through several hops from source to destination. Usually, end-to-end delay is the major concern. So in ad hoc scenarios, we choose end-to-end delay as the QoS metric.

### 2.1 Cross-layer Framework of QoS Provision

Figure 1 shows the framework of our scheme. The application notifies the middleware that it wishes to set up a flow between two end hosts with certain QoS specification. The middleware decides an appropriate service class for each application dynamically according to the performance of the flow and its QoS requirement. Middleware contains three components: *Classifier*, *Priority Adaptor* and *QoS Monitor*.

For WLAN scenario, *QoS Monitor* is a *Bandwidth Monitor* which measures the bandwidth for applications. In [2], bandwidth is estimated by the information coming from the

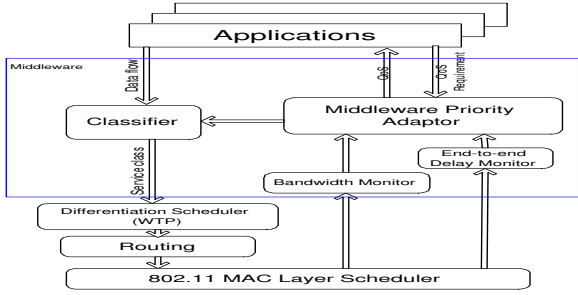


Figure 1. Cross-layer framework of middleware adaptation in WLAN or ad hoc scenario

MAC. Since we use the standard wireless device in our testbed evaluation, we do not want to modify the MAC layer implementation. Therefore, we implemented our bandwidth estimator in the kernel of the operating system. This implementation allows our *Bandwidth Monitor* be device-independent. According to [2], the bandwidth then can be derived as  $BW = \frac{S}{t_r - t_s}$ , where  $S$  is the size of the packet,  $t_s$  is the time-stamp that the packet is ready at the MAC layer, and  $t_r$  is the time-stamp that an ACK has been received.  $t_r - t_s$  includes the channel busy and contention time.

For multi-hop adhoc scenario, *QoS Monitor* is *Delay Monitor* which monitors the end-to-end delay of each service class and notifies the observed changes and QoS violations to the *Adaptor*. It measures the average round trip delay incurred to deliver packets. The sender attaches time stamps when sending the packets to the destination. As the sender gets ACK from the destination, it retrieves the sending time stamp, and compares it with the current time stamp, so that a sender can obtain the round-trip delay  $d_i$  for packet  $i$ . We take an average of  $N$  round-trip delay measurements ( $d_1, d_2, \dots, d_N$ ), and have  $d_{avg} = \frac{1}{2N} \sum_{i=1}^N d_i$  as the measured value to estimate end-to-end delay.

*Priority Adaptor* determines the priority of the packets according to delay and QoS specification. The *Classifier* marks the packets with their corresponding service classes according to their priority values. At network layer, the *Differentiated Scheduler* selects a packet to transmit. It performs packet-level QoS enforcement, allocates bandwidth for different flows and provides waiting time priority (WTP) scheduling. At MAC layer, we adopt IEEE 802.11a/b/g protocols.

## 2.2 Middleware Model

The middleware priority adaptor will be modelled using feedback control theory. Our goal is to derive a relatively stable adaptor. Due to the large system and observation noises, we choose the PI (Proportional and Integral) con-

troller for priority adaptation, which is the function of *Middleware Adaptor*. Block diagram of the system is shown in Figure 2.

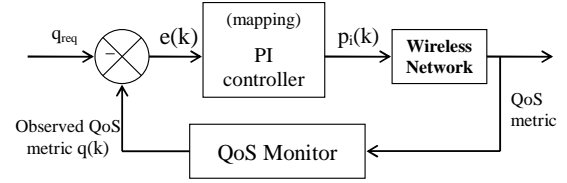


Figure 2. Control loop of the system

Let  $q(k)$  be a QoS metric (bandwidth or end-to-end delay) at time  $k$ , and  $q_{req}$  be the QoS requirement. So  $e(k) = q(k) - q_{req}$  is the QoS error, which is the difference between desired and actual QoS levels. Let  $p_i(k)$  be the packet priority at time  $k$  for application  $i$ . The PI controller is in the form that:

$$p_i(k+1) = p_i(k) + (k_p + k_i)(q(k) - q_{req}) + k_i(q(k-1) - q_{req})$$

where  $k_p$ ,  $k_i$  are proportional and integral controller parameters, respectively. The coefficients  $k_p$ ,  $k_i$  map the unit difference between performance and packet priority. So their units should be  $\frac{1}{QoS \text{ measurement unit}}$ , where  $QoS \text{ measurement unit}$  is unit of delay or bandwidth.

## 2.3 WTP Scheduler at Network Layer

To enforce the service differentiation, we adopt waiting time priority (WTP) scheduler at network layer. [19] introduces the proportional differentiation model in wireline networks, which offers relative differentiated services between a small number of service classes with different service quality in queueing delay and packet losses. [20] addresses the proportional service differentiation in terms of throughput in WLANs. The proportional delay differentiation (PDD) model has the property that for any pair of classes  $i$  and  $j$ , in the time interval  $(t, t + \tau)$ ,

$$\frac{\bar{d}_i(t, t + \tau)}{\bar{d}_j(t, t + \tau)} = \frac{\delta_i}{\delta_j} \quad (1)$$

where  $\bar{d}_i(t, t + \tau)$  is the average delay for class  $i$ , and  $\delta_i$  is the delay differentiation parameter (DDP) for class  $i$ . The gap between absolute QoS guarantee and relative differentiation is bridged by dynamic service class selection [21]. In this paper, we use waiting time priority (WTP) scheduler to approximate PDD model. In the WTP scheduling algorithm, the *normalized head waiting time* of class  $i$  at time  $t$  is defined as  $\tilde{w}_i(t) = w_i(t)/\delta_i$ , where  $w_i(t)$  is the actual waiting time of the head packet in service class  $i$ . The WTP scheduler selects the packet from class  $j$  with the maximum normalized head waiting time  $j = \arg \max_{i \in B(t)} \tilde{w}_i(t)$

to serve, where  $B(t)$  stands for the set of classes at time  $t$  waiting to be served. [22] shows that as the utilization converges to 1, waiting time priority (WTP) scheduler is one of the algorithms to approximate PDD in wired networks under the assumption of Pareto or Poisson arrivals.

To provide the end-to-end delay support for multimedia applications on wireless networks, we propose a priority based delay differentiated service at network layer, and priority adjustment algorithm at middleware layer to achieve the end to end delay guarantee in IEEE 802.11 wireless ad hoc environment. In a multimedia stream flow, we assume the packet size is fixed. However, in wireless ad hoc environment, due to the contention overhead at MAC layer, even if the packet size is fixed, the time to forward a packet is not fixed. For multimedia flows over IEEE 802.11 ad hoc networks, we need to assume more general packets inter-arrival distributions, since the Poisson arrivals and Pareto distributions are not valid. No previous work shows that when the packet arrival pattern is not Pareto or Poisson, the WTP scheduler still approximates PDD model. We show in the appendix that for data intensive and interleaved flows, such as multimedia flows, the PDD property holds. So the ratio of average delay for any class pair  $i$  and  $j$  converges to PDD model as the utilization  $u$  tends to go to 100%

$$\frac{\bar{d}_i^{WTP}}{\bar{d}_j^{WTP}} \rightarrow \frac{\delta_i}{\delta_j} \quad \text{as } u \rightarrow 1 \quad (2)$$

In our implementation of WTP scheduler, a packet in service class  $i$  is attached with a class parameter  $p_i$ , which is interpreted as priority of service class  $i$ .  $p_i$  is determined dynamically by the middleware. When a packet needs to be transmitted, the network layer scheduler selects the packet  $p$  with the maximum normalized waiting time

$$p = \arg \max_{p \in P(t)} w_p(t) p_i \quad (3)$$

where  $P(t)$  is the set of packets waiting to be served in time  $t$  and  $w_p(t)$  is the waiting time of a packet  $p$ . The application packet with a larger priority  $p_i$  is assigned more service. In Appendix, we show that under the assumption that the flows are data-intensive, and interleaved with each other, the WTP scheduler given by equation (3) approximates the PDD model. In our WTP model,  $p_i = \frac{1}{\delta_i}$ . We notice that *Proposition 1* is consistent with the statement made in [22] that the ratio of average delay for any class pair of  $i$  and  $j$  converges to PDD model as the utilization  $u$  tends to go to 100%, since data-intensive flows drive the utilization tends to 100%. With *Proposition 1*, we have the intuition that by increasing the priority of a multimedia flow, the delay ratio of the multimedia flow to best-effort flows will decrease. Therefore, we can control the average end-to-end delay or bandwidth by adjusting the priority of the multimedia flow.

## 3 Experimental Study

### 3.1 Goal of Validation

As stated in the introduction, the middleware adaptation should be light-weighted, flexible and correct. By light-weight, we mean small cost will be induced in terms of line of codes and memory usage when the middleware is loaded into the target system. Flexibility means the middleware design does not require to modify the MAC layer and is resilient to various changes coming from the operating system and the wireless networking system. Finally, correctness means that the middleware can keep QoS within certain thresholds of the required level. All experiments of our middleware validate these three properties.

We here implemented our control-based middleware scheme in C++ and Java. The *Monitor* and *Scheduler* are implemented as kernel modules in Linux. Their total size is about 600 lines of C++ code. All java and C++ components communicate via *proc* file system. The total size of the *Adaptor* and *Classifier* is approximately 500 lines of Java code. However, the core algorithm of the *Adaptor* takes only about 100 lines of Java code. Thus, our middleware scheme can fit into the system at very low cost.

### 3.2 Experimental Setup

Our testbed includes five laptops: two IBM T41 laptops, two G41 laptops and one T42 laptop. Each laptop is equipped with an IEEE 802.11b Lucent Gold wireless card. Laptops are connected in ad-hoc mode. All laptops run on Fedora Core 2 (Kernel 2.6.5) and are installed with Java Media Framework (JMF) to record and play audio. Audio in JMF is built essentially on top of UDP.

### 3.3 Metrics

We use bandwidth and delay as metrics for our experiments. Our middleware will take end-to-end bandwidth and end-to-end delay as the input for adaptation. We also investigate the impact of these two metrics on our middleware.

### 3.4 Scenarios

We conduct experiments in two modes: WLAN and Ad-hoc mode. In WLAN mode, we test three scenarios: uplink scenario, downlink scenario and mobility scenario. By uplink, we mean data will go from nodes to the base station and downlink means data will go from the base station to nodes. Figure 3.(a) shows the topology used in WLAN mode. The base station is the T42 laptop and two nodes are T41 laptops. Table 1 shows all parameters for each scenario.

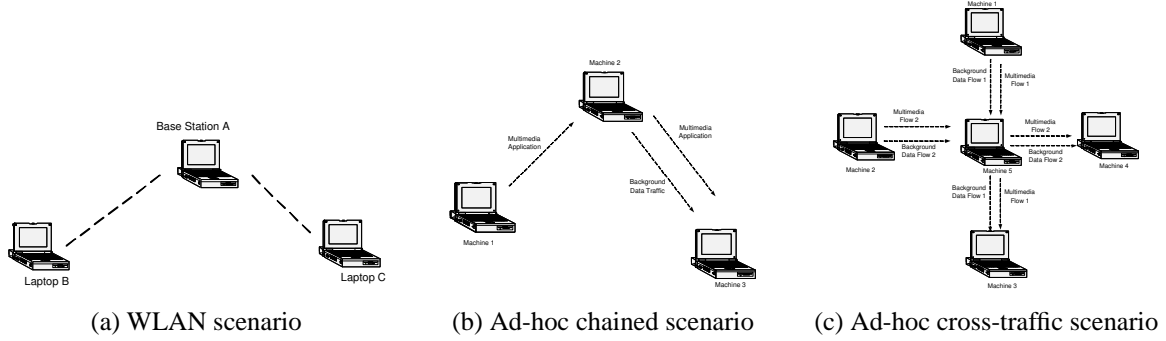


Figure 3. Experimental scenarios

WLAN Mode			
Scenario	Capacity	Audio	UDP
1) Downlink	1M	800K	800K
2) Uplink	2M	1.7M	1.7M
3) Mobility	2M	800K	1M
Ad-hoc Mode			
Scenario	Capacity	Voice	UDP
4) Chain-traffic	2M	32K	1M
5) Cross-traffic	2M	32K	200K

Table 1: Summary of scenarios

In ad-hoc mode, we test two scenarios: chained scenario and cross-traffic scenario. The topology for the chained scenario and the cross traffic scenario are shown in Figure 3.(b) and Figure 3.(c). Table 1 summarizes parameters for each scenario.

### 3.5 Results

In this section, we show the results of all scenarios.

#### Scenario 1: WLAN Downlink scenario<sup>1</sup>

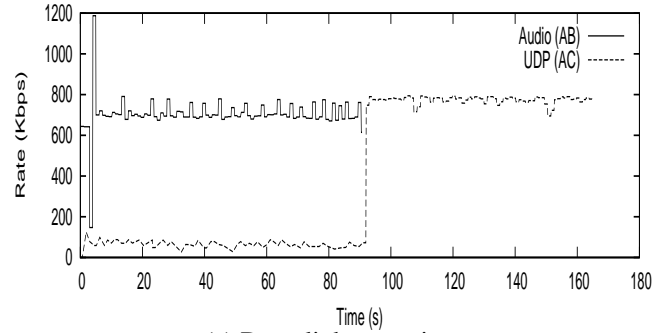
As shown in Figure 4.(a), our middleware scheme works in this scenario. During the first 5 seconds, both streams compete for the bandwidth and thus, the rate of Audio stream is not stable. After that the *Adaptor* adjusts the priority of the Audio stream to meet the bandwidth requirement. At 90th second when audio stream stops, the UDP stream can transmit at its original rate normally.

In this scenario, our middleware responds very quickly to the changes of the networking condition. Many small spikes in Figure 4.(a) show that whenever a change of bandwidth occurs, our scheme can detect and adjust the priority very quickly. Therefore, in this scenario, flexibility is shown very clearly.

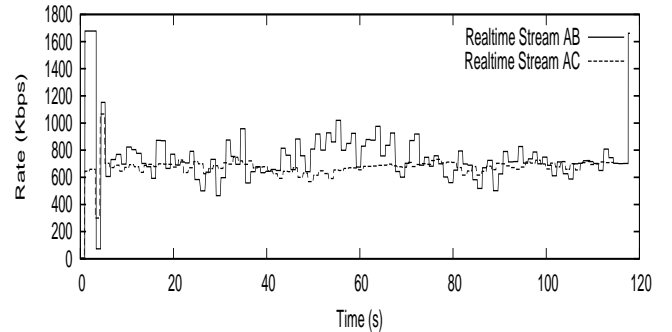
Our scheme also shows a reasonable correctness of QoS adaptation. The rate of the audio stream is guaranteed within a certain level of the requested rate as shown in figure 4.

#### Scenario 2: WLAN Uplink scenario

<sup>1</sup>In WLAN scenarios, bandwidth of a one-hop wireless link implies delay. Therefore, we use bandwidth as the QoS metric.



(a) Downlink scenario



(b) Uplink scenario

Figure 4. Bandwidth of Audio and UDP stream

As shown in Table 1 and Figure 3.(a), in this scenario node *B* sends an 1.7Mbps audio stream to *A* and node *C* sends an 1.7Mbps UDP stream to *A*. Since the network capacity is only 2Mbps and both streams send simultaneously, their requested rate cannot be satisfied. As shown in Figure 4.(b), during the first few seconds, both streams experience a big fluctuation in their bandwidth due to the lack of total bandwidth. Although the *PriorityAdaptor*, resided in node *B* and node *C*, adjust the priority due to the changes of bandwidth, it does not help. The reason is that both priority adaptations happen locally, i.e. the change

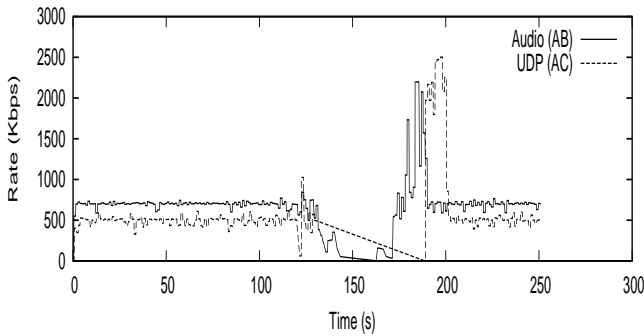
of priority in  $B$  does not affect  $C$  and vice versa. The competition between two streams comes up as expected: both streams share the total bandwidth, i.e. each gets half of the total capacity.

We can see that in such a case, global communication between nodes, such as MAC layer differentiation scheduler [14], is needed.

### Scenario 3: WLAN Mobility scenario (Downlink)

In this scenario, the audio node is moved far away from the base station during the experiment. We refer to this node as  $B$ .

As shown in Figure 5, during the first 100 seconds, both streams transmit at its request rate. After that,  $B$  starts moving. At around the time 150s,  $B$  experiences the decreasing of bandwidth which triggers the *Adaptor* to increase the priority of its audio stream. After 150th second, the priority of the audio stream is increased quickly and eventually at around 200th second, the bandwidth requirement is met.



**Figure 5. Bandwidth of Audio and UDP stream (Mobility)**

As the node moves, the bandwidth is very hard to estimate due to large loss rate and unpredictable delay. Consequently, it is very difficult for any admission schemes to deal with mobility. Thus, mobility is also hard to deal for any protocols that require exchanging global information between nodes. Our scheme with its light-weight, however, can adapt to the mobility within a reasonable of time since it does not require neither any admission control mechanism nor exchanging global information between nodes.

### Scenario 4: Ad Hoc Chained Scenario <sup>2</sup>

Figure 3.(b) shows the experiment setup for this scenario. In the experiment, Machine 1 sends multimedia traffic (audio application of sampling rate 32KBps) to Machine 3, where Machine 2 serves as a router of the multimedia traffic. At the same time, Machine 2 generates UDP background traffic to Machine 3, with data rate 1Mbps. The

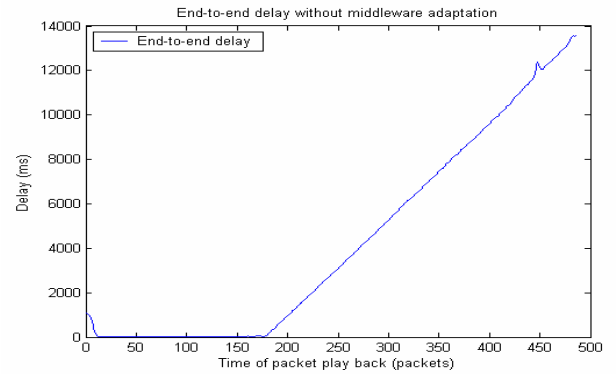
<sup>2</sup>Since in multi-hop ad hoc scenarios, neighboring links share the bandwidth capacity, bandwidth of a single hop is meaningless. Therefore, we use end-to-end delay as the QoS metric.

wireless link capacity is 2Mbps.

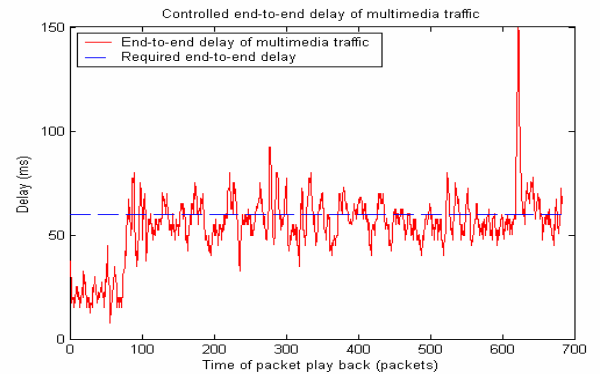
First, we do the experiment where no middleware adaptation is adopted. In this case, the background UDP traffic starts after the multimedia application sends 170 packets. Under the background traffic at the rate of 1Mbps, the end-to-end delay of multimedia application tends to be very large (Figure 6.(a)).

The targeted end-to-end delay is 60 millisecond ( $ref_{delay} = 60$ ). Figure 6.(b) shows the resulting end-to-end delay under the middleware priority adaptation. In this case, the background data traffic starts after the multimedia application sends around 60 packets.

The experiment shows that the middleware adaptor is able to converge quickly the end-to-end delay of a multimedia application to a desired level, when there exist other applications which compete with the multimedia application for the network resources in wireless adhoc environment.



(a) No priority adaptation



(b) With Priority Adaptation by PI controller

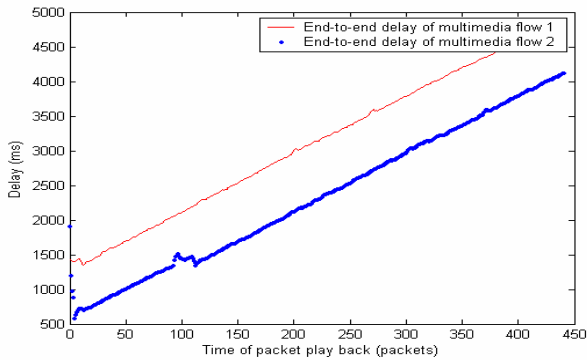
**Figure 6. Chained scenario**

### Scenario 5: Ad Hoc Cross Traffic Scenario

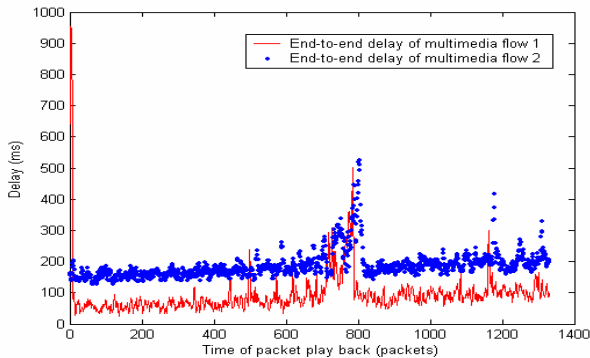
Figure 3.(c) shows the experiment setup for scenario 2. In this case, Machine 1 sends multimedia flow 1 and background data flow 1 to Machine 3, Machine 2 sends multimedia flow 2 and background data flow 2 to Machine 4.

Machine 5 serves as the router. The background data rate is 200Kbps.

We set the expected end-to-end delay (delay reference) of multimedia flow 1 be 100ms, and the end-to-end delay of multimedia flow 2 be 200ms. With no priority adaptation, the end-to-end delay tends to very large (Figure 7.(a)). With middleware priority update, the end-to-end delays of multimedia flows are shown in Figure 7.(a)). We notice the spikes in Figure 7.(b). When network load turns to heavier, some packets suffer very large delay. We adjust priority of packets when we observe the large delay, so that the later packets will encounter small delay. The spikes are formed in this way that some packets get large delay in a very short time.



(a) No priority adaptation



(b) With Priority Adaptation by PI controller

**Figure 7. Cross traffic scenario**

## 4 Discussions

Our experimental evaluation shows that the middleware adaptation is needed and efficient in many scenarios to provide QoS support for traffic which has QoS requirements in wireless networks. In some cases, we need middleware adaptation and MAC layer QoS support to work together. For example, in the uplink WLAN scenario, only middle-

ware adaptation is not capable to provide satisfied bandwidth requirement.

Our middleware adaptation framework is a light-weighted and efficient solution, which was overlooked in previous research. With the aid of middleware support, traffic with QoS requirements will be easier to control in order to meet the QoS requirements.

## References

- [1] G.-S. Ahn, A. T. Campbell, A. Veres, and L.-H. Sun, "Supporting service differentiation for real-time and best-effort traffic in stateless wireless ad hoc networks (swan)," *IEEE Transactions on Mobile Computing*, vol. 1, no. 3, pp. 192–207, 2002.
- [2] S. H. Shah, K. Chen, and K. Nahrstedt, "Dynamic Bandwidth Management for Single-hop Ad Hoc Wireless Networks," in *Percom*, 2003.
- [3] Y. Yang and R. Kravets, "Contention-aware admission control for ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 4, pp. 363–377, 2005.
- [4] C. Curescu, "Utility-based optimisation of resource allocation for wireless networks," *Ph.D Thesis*, 2005.
- [5] J. M. Capone and I. Stavrakakis, "Delivering diverse delay/dropping qos requirements in a tdma environment." in *MOBICOM*, 1997, pp. 110–119.
- [6] W.-H. Liao, Y.-C. Tseng, and K.-P. Shih, "A tdma-based bandwidth reservation protocol for qos routing in a wireless mobil ad hoc network," in *IEEE ICC*, 2002.
- [7] "IEEE 802.11 standard, wireless LAN medium access control (mac) and physical layer (phy) specifications."
- [8] C. Coutras, S. Gupta, and N. B. Shroff, "Scheduling of real-time traffic in ieee 802.11 wireless lans," *Wireless Network*, vol. 6, no. 6, pp. 457–466, 2000.
- [9] J. L. Sobrinho and A. S. Krishnakumar, "Quality-of-service in ad hoc carrier sense multiple access wireless networks," *IEEE JSAC*, vol. 17, no. 8, pp. 1353–1368, August 1999.
- [10] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11," in *INFOCOM*, 2001, pp. 209–218.
- [11] A. Banchs, M. Radimirsch, and X. Perez, "Assured and expedited forwarding extensions for IEEE 802.11 wireless LAN," in *IEEE/IFIP IWQoS*, 2002, pp. 237–246.

- [12] S. T. Sheu and T. F. Sheu, "A bandwidth allocation/sharing/extension protocol for multimedia over IEEE 802.11 ad hoc wireless lans," *IEEE JSAC*, vol. 19, no. 10, pp. 2065–2080, October 2001.
- [13] W. Yuan, K. Nahrstedt, S. Adve, D. Jones, and R. Kravets, "Design and evaluation of a cross-layer adaptation framework for mobile multimedia systems," in *SPIE/ACM Multimedia Computing and Networking Conference*, 2003.
- [14] Y. Xue, K. Chen, and K. Nahrstedt, "Achieving Proportional Delay Differentiation in Wireless LAN via Cross-Layer Scheduling," in *Journal of Wireless Communications and Mobile Computing, Special Issue on Emerging WLAN Technologies and Applications*, 2004.
- [15] T. Abdelzaher, K. G. Shin, and N. Bhatti, "Performance guarantees for web server endsystems: A control-theoretical approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, 2002.
- [16] N. Gandhi, S. Parekh, J. Hellerstein, and D. Tilbury, "Feedback control of a lotus notes server: modeling and control design," in *American Control Conference*, 2001.
- [17] B. Li and K. Nahrstedt, "A control-based middleware framework for quality of service adaptations," *IEEE J. Select. Areas Commun.*, September 1999.
- [18] S. H. Shah, K. Chen, and K. Nahrstedt, "Dynamic bandwidth management for single-hop ad hoc wireless networks," *ACM/Kluwer Mobile Networks and Applications (MONET) Journal, Special Issue on Algorithmic Solutions for Wireless, Mobile, Ad Hoc and Sensor Networks*, vol. 10, no. 1, 2005.
- [19] C. Dovrolis and P. Ramanathan, "A case for relative differentiated services and the proportional differentiation model," *IEEE Network*, vol. 13, no. 5, 1999.
- [20] Q. Xue and A. Ganz, "Proportional service differentiation in wireless lans using spacing-based channel occupancy regulation," in *ACM Multimedia*, November 2004.
- [21] C. Dovrolis and P. Ramanathan, "Dynamic class selection: From relative differentiation to absolute QoS," in *Proceedings of ICNP*, Nov. 2001.
- [22] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," in *SIGCOMM*, 1999, pp. 109–120.

## Appendix

Assume the class  $i$  is assigned with class parameter  $p_i$ , the scheduling algorithm in (3) is equivalent to select the packet in class  $i$  to serve if  $i = \arg \max_{j \in B(t)} w_j(t) p_j$ .

Let  $n_i(t, \tau)$  be the number of packets of class  $i$  received and queued by network layer WTP scheduler in time period  $[t, t + \tau]$ , and  $m_i(t, \tau)$  be the number of packets of class  $i$  served by the scheduler in time period  $[t, t + \tau]$ . Denote  $a_i(t)$  be the arrival time of the last served packet from service class  $i$  before time  $t$ ,  $b_i(t)$  be the arrival time of the packet which is first served from service class  $i$  after time  $t$ , and  $s_i(t)$  be the service begin time of the last packet served from class  $i$  before time  $t$ . Consider any pair of service classes  $i$  and  $j$ , ( $i, j \in B(t)$  and  $i \neq j$ ), the packets from classes  $i$  and  $j$  interweave with each other.

**Lemma 1:** For data-intensive and interleaved flows, we have  $(t - a_i(t))p_i \simeq (t - a_j(t))p_j$ .

**Proof:**

By the scheduling algorithm (3), we have

$$(t - a_i(t))p_i > (t - b_j(t))p_j \quad (4)$$

and

$$(t - a_j(t))p_j > (t - b_i(t))p_i \quad (5)$$

If  $s_i(t) > s_j(t)$ , then last packet served by the scheduler is from class  $i$ , we have:

$$(t - a_j(t))p_j > (t - a_i(t))p_i \quad (6)$$

Combine equation (4) and (6), we have

$$(t - a_j(t))p_j > (t - a_i(t))p_i > (t - b_j(t))p_j \quad (7)$$

For data-intensive and interleaved flows, the packet arrivals from class  $j$  are close enough, so that  $a_j(t) \simeq b_j(t)$ . Similar to the packet arrivals from packet  $i$ , we have  $a_i(t) \simeq b_i(t)$ . Therefore,  $(t - a_j(t))p_j \simeq (t - b_j(t))p_j$  and  $(t - a_i(t))p_i \simeq (t - b_i(t))p_i$ . Thus, by equation (7), we have the approximation that  $(t - a_i(t))p_i \simeq (t - a_j(t))p_j$ .

Similarly, if  $s_i(t) < s_j(t)$ , we can get the same result.

**Lemma 2:** For data-intensive and interleaved flows, during the period  $[t, t + \tau]$ ,  $\left(1 - \frac{m_i(t, \tau)}{n_i(t, \tau)}\right) p_i \simeq \left(1 - \frac{m_j(t, \tau)}{n_j(t, \tau)}\right) p_j$ .

**Proof:**

By Lemma 1, we have

$$(t + a_i(t))p_i = (t + a_j(t))p_j \quad (8)$$

and

Therefore,

$$\frac{\bar{d}_i(t, \tau)}{\bar{d}_j(t, \tau)} = \frac{p_j}{p_i} \quad (19)$$

$$(t + \tau + a_i(t + \tau))p_i = (t + \tau + a_j(t + \tau))p_j \quad (9)$$

From equation (8) and (9), we have

$$\frac{a_j(t + \tau)p_j - a_i(t + \tau)p_i}{p_j - p_i} = t + \tau \quad (10)$$

and

$$\frac{a_j(t)p_j - a_i(t)p_i}{p_j - p_i} = t \quad (11)$$

$n_i(t, \tau)$ , the arrival packets of class  $i$  being queued during  $[t, t + \tau]$  and the packets being served  $m_i(t, \tau)$  satisfies

$$\frac{n_i(t, \tau)}{\tau} \simeq \frac{m_i(t, \tau)}{a_i(t + \tau) - a_i(t)} \quad (12)$$

Substitute equation (12) to (10) - (11), we have

$$\left(1 - \frac{m_i(t, \tau)}{n_i(t, \tau)}\right)p_i \simeq \left(1 - \frac{m_j(t, \tau)}{n_j(t, \tau)}\right)p_j \quad (13)$$

**Proposition 1:** For data-intensive and interleaved flows, with the WTP scheduler described in (3), the delay at each hop approximates proportional delay differentiation (PDD) model.

**Proof:**

Based on equation (13), we have

$$\frac{n_i(t, \tau) - m_i(t, \tau)}{n_j(t, \tau) - m_j(t, \tau)} = \frac{p_j n_i(t, \tau)}{p_i n_j(t, \tau)} \quad (14)$$

Let  $\lambda_i(t, \tau)$  be the average arrival rate of class  $i$  within the time period  $[t, t + \tau]$ . Then  $\lambda_i(t, \tau) = \frac{n_i(t, \tau)}{\tau}$ . By Little's Law, we know that

$$\lambda_i(t, \tau) \times \bar{d}_i(t, \tau) = \bar{L}_i(t, \tau) \quad (15)$$

where  $\bar{d}_i(t, \tau)$  and  $\bar{L}_i(t, \tau)$  are average delay and average queue length of class  $i$  packets within the time period  $[t, t + \tau]$ . Since  $n_i(t, \tau) - m_i(t, \tau)$  is the number of backlogged packets from class  $i$ , the ratio of average queue length is given by

$$\frac{\bar{L}_i(t, \tau)}{\bar{L}_j(t, \tau)} = \frac{n_i(t, \tau) - m_i(t, \tau)}{n_j(t, \tau) - m_j(t, \tau)} \quad (16)$$

From (14), (15) and (16) we have

$$\frac{\bar{d}_i(t, \tau)\lambda_i(t, \tau)}{\bar{d}_j(t, \tau)\lambda_j(t, \tau)} = \frac{n_i(t, \tau) - m_i(t, \tau)}{n_j(t, \tau) - m_j(t, \tau)} = \frac{p_j n_i(t, \tau)}{p_i n_j(t, \tau)} \quad (17)$$

Thus,

$$\frac{\bar{d}_i(t, \tau)n_i(t, \tau)}{\bar{d}_j(t, \tau)n_j(t, \tau)} = \frac{p_j n_i(t, \tau)}{p_i n_j(t, \tau)} \quad (18)$$