

Application of Network Layer Mechanisms for Service Differentiation in 802.11 WLANs: Implementation and Experience

Vasilios A. Siris and George Stamatakis

Institute of Computer Science, Foundation for Research and Technology - Hellas (FORTH)

P.O. Box 1385, GR 711 10 Heraklion, Crete, Greece

Tel.: +30 2810 391726, fax: +30 2810 391601, email: {vsiris,gstam}@ics.forth.gr

Abstract— In this paper we present the design, implementation, and experience in using network layer mechanisms for service differentiation in 802.11 Wireless LANs (WLANs). In particular, we use the Class-Based Weighted Fair Queuing (CBWFQ) mechanism with weights that are dynamically adjusted based on the achieved throughput of the wireless stations and their physical layer transmission rate. The scheme was implemented in a Linux-based testbed, and experiments show that it can effectively support throughput differentiation, and deal with important WLANs problems, such as unfairness due to location-dependent channel errors, uplink-downlink unfairness, and performance degradation for the whole WLAN when a station transmits at a small rate. The experiments involved best-effort traffic, as well as streaming video and voice-over-IP.

Keywords: fairness, bursty errors, multirate

I. INTRODUCTION

In the last few years, wireless networks based on IEEE 802.11 have been experiencing a tremendous growth. Their success can be attributed to the utilization of unlicensed spectrum, interoperability through standard implementations, low cost, and simple installation and management. The increasing use of wireless networks by demanding applications and users with different requirements has created the need for service differentiation, while efficiently utilizing the shared wireless channel. Supporting service differentiation in wireless networks is hindered by several factors, such as location-dependent channel errors, uplink-downlink unfairness, and performance reduction due to the multirate physical layer. Nevertheless, a number of solutions to these problems have been proposed at various network layers. Few of these solutions, however, can be implemented in a feasible and effective way without requiring alternation of the 802.11 MAC layer.

In this paper we investigate a dynamic Class-Based Weighted Fair Queuing (CBWFQ) scheme, introduced initially in [1], that supports throughput differentiation, while improving fairness and enhancing the efficient utilization of the wireless channel. The current paper contains a discussion of the fairness issues that motivate the proposed scheme, a discussion of its operation and implementation, and present testbed results involving streaming video and voice-over-IP traffic. The proposed mechanism is based on a definition of

fairness inherently different from the fairness objective of the IEEE 802.11 MAC protocol. In particular, fairness is expressed in terms of the channel occupation time, which suits best the multirate physical layer of 802.11. The proposed scheme is comprised of two algorithms that utilize cross-layer information to periodically adjust the CBWFQ weights in a way that collectively deals with significant issues of WLANs, such as location-dependent channel errors, uplink downlink unfairness, and unfairness due to the multirate operation of 802.11. The first algorithm monitors the throughput achieved by each node and adjusts weights in order to compensate nodes that suffered losses due to channel errors, or did not have packets to transmit due to traffic burstiness. The second algorithm incorporates transmission rate information in the weight adjustment procedure, so that nodes occupy the channel for a time duration proportional to their weights.

The dynamic CBWFQ scheme is a network layer mechanism whose implementation does not require changes to the 802.11 MAC layer. Its deployment can be done either in a control station, that interconnects one or more access points with the wired network, or as a service running on an access router. The former approach enables the proposed scheme to work with legacy 802.11 access points. In this paper we focus on the latter deployment approach, and present experiments illustrating its effectiveness in supporting service differentiation in the presence of different traffic types, including best-effort traffic, streaming video, and voice-over-IP.

The rest of the paper is organized as follows. In Section II we discuss fairness issues in 802.11. In Section III we describe our service differentiation mechanism. In Section IV we describe the testbed implementation and evaluation scenarios, and analyze the experimental results. In Section V we review related work and in Section VI we present our conclusions and identify related ongoing and future work.

II. FAIRNESS IN IEEE 802.11 AND RELATED ISSUES

The distributed coordination function (DCF) in IEEE 802.11 is responsible for controlling access to the shared wireless medium. DCF is a random access scheme, based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

protocol, that guarantees an equal long term channel access probability to all wireless stations.

DCF is the root of significant unfairness issues in 802.11 networks. At first, transmission failures due to location-dependent bursty channel errors prevent the wireless stations from utilizing their share of transmission attempts and thus cause unfair bandwidth distribution. Another cause of unfairness is the so-called “channel capture effect”, where after a collision the node that successfully transmitted a packet has a higher probability of transmitting subsequent packets, compared to stations whose packets collided.

In infrastructure-based WLANs the access point serves all downlink traffic, but there is no anticipation in DCF to increase its transmission priority. Moreover, in case of buffer overflow at the access point, downstream flows will suffer data packet losses while upstream flows will suffer acknowledgment losses. However, a data packet loss results in TCP’s congestion window being reduced while loss of an acknowledgment has no significant influence on the congestion window, due to the cumulative acknowledgment feature of TCP. These two phenomena cause a significant uplink-downlink unfairness problem as described in [2].

Another significant issue observed in [3], and termed “performance anomaly”, is that the multirate physical layer of 802.11 and the equal number of transmission attempts guaranteed by DCF, result in some nodes occupying the channel for longer time periods than others. The 802.11b physical layer defines four possible transmission rates: 1, 2, 5.5 and 11 Mbps. A node transmitting with a bit rate lower than 11 Mbps will acquire the channel longer than a node transmitting at 11 Mbps, in order to transmit the same amount of data. This leads to an unfair distribution of channel occupation time that causes significant degradation of the aggregate throughput.

A fairness scheme that suits better the multirate physical layer of 802.11 could be expressed as a *weighted fairness scheme, in terms of successful transmission and reception time shares* ([4], [5]). If we assure that all nodes are granted their transmission time share as determined by their weights, the performance of nodes with high quality channels would not be influenced by the channel quality of the other nodes. Likewise, nodes with poor channels are guaranteed their own time share. Furthermore, the sum of reception time shares of all nodes corresponds to the transmission time of the access point. By granting each node its reception time share, according to its weight, we can differentiate service in the downstream direction and address the problem of uplink-downlink unfairness.

III. DYNAMIC CBWFQ

Direct implementation of the fairness scheme described above implies a radical redesign of the 802.11 MAC layer. In order to avoid changing the MAC layer, we adopted a dynamic CBWFQ mechanism that incorporates transmission rate information in order to approximate the behavior of the proposed fairness scheme.

In wired networks, CBWFQ (Class-Based Weighted Fair Queuing) has long been a popular paradigm for achieving fair capacity sharing and bounded delays. In a CBWFQ scheme, each packet is assigned to a *traffic class* based on information included in the packet’s header. Each traffic class i is given a weight w_i and for any time interval $[t_1, t_2]$ during which there is no change in the set of backlogged classes $B(t_1, t_2)$ the throughput $T_i(t_1, t_2)$ achieved by class i satisfies

$$\forall i, j \in B(t_1, t_2), \quad \left| \frac{T_i(t_1, t_2)}{w_i} - \frac{T_j(t_1, t_2)}{w_j} \right| = 0.$$

CBWFQ provides isolation among different classes, hence a class’s minimum throughput guarantee is unaffected by the behavior of other classes.

There are significant issues concerning the application of the CBWFQ mechanism in wireless networks. CBWFQ assumes that the channel is error-free or, at the very least, errors are not location-dependent. This assumption does not hold in wireless networks. Another unique characteristic of wireless networks is that both uplink and downlink transmissions share the same wireless channel. The uplink traffic, however, cannot be restrained by the CBWFQ mechanism running at the access point, since each host schedules its packets independently. Furthermore, CBWFQ cannot provide fairness and service differentiation for bursty data traffic, when the network utilization is not very high. Hence, when a node has no packets to send, e.g. due to bursty packet arrivals, the CBWFQ mechanism schedules packets from other nodes, thus some nodes could achieve higher throughput than the minimum their weight guarantees. The above “absent” node, however, cannot reclaim the service it relinquished when it becomes backlogged.

Next we present the architecture of our dynamic CBWFQ mechanism that addresses all the above issues, and involves the dynamic adjustment of weights utilizing cross-layer information. The dynamic CBWFQ mechanism is composed of three parts: i) a service differentiation mechanism for controlling both downlink and uplink traffic, ii) a throughput compensation algorithm that addresses the issues of location-dependent bursty channel errors and bursty packet transmission, and iii) a performance improvement algorithm that incorporates transmission rate information for adjusting weights.

A. Service Differentiation Mechanism

The service differentiation mechanism provides per node throughput differentiation in both the downlink and the uplink, using the CBWFQ mechanism of Fig. 1. Each wireless node is related with two packet classes, the DATA and the ACK class. The DATA class matches the data packets (TCP or UDP) destined to the node, while the ACK class matches the TCP acknowledgment packets destined to it. Every class is assigned a separate queue for its packets and the bandwidth is shared among them according to their weights. If demand for resources is low, the unutilized bandwidth will be shared among the backlogged nodes in proportion to their weights.

The queues corresponding to each node can be created once a node becomes associated with an access point. Additionally,

recent activity information can be used to remove queues that correspond to dormant nodes or nodes who left without disassociating themselves.

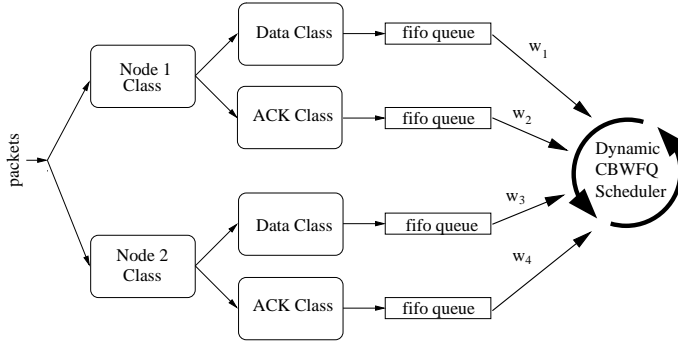


Fig. 1. Service differentiation mechanism.

For TCP, assigning acknowledgment packets destined to a node into a separate queue provides a way to control the node's uplink traffic, by varying the transmission rate of acknowledgment packets which subsequently affect the transmission rate of data packets at the sending node. Indeed, experiments suggest that the TCP uplink rate is approximately proportional to the rate of ACKs. Although there is no corresponding method for controlling uplink UDP traffic in general, our method can be applied when some rate control over UDP is used, as in video streaming and DCCP (Datagram Congestion Control Protocol).

The mechanism shown in Fig. 1 can support per node throughput differentiation for downlink and uplink (TCP) traffic. It can be easily extended to support multiple classes per node, thus allowing the support of different applications running in the same node.

It must be noted here that packets transmitted to nodes with poor channel quality could introduce head of line blocking at the FIFO queue of the network interface card (NIC) due to MAC layer retransmissions. This problem is mitigated by our performance improvement algorithm that restrains potential transmission failures by decreasing the weight of nodes with reduced transmission rate due to poor channel conditions and hence the number of packets that can cause head of line blocking.

B. Throughput Compensation Algorithm

The effectiveness of the CBWFQ scheme with static weights is limited due to the existence of location-dependent bursty channel errors. This can be addressed by adjusting the CBWFQ weights according to the actual throughput achieved. Such an approach can achieve long term fairness by giving more service to a node that previously experienced an erroneous channel or to a node previously absent, i.e. did not have packets to transmit. Indeed, accounting absence as loss of service improves performance in the case of bursty data traffic, since a node would obtain a higher weight when it is transmitting a burst of packets due to its lower (zero) throughput when it was not transmitting; related experimental results will be presented in Section IV.

Weights are adjusted in time periods of equal length, based on the amount of service lost or gained. The amount of service lost or gained by class i up to period n is

$$D_i[n] = D_i[n-1] + (achieved_r_i[n] - target_r_i[n]), \quad (1)$$

where $achieved_r_i[n]$ is the rate achieved by class i during period n and $target_r_i[n]$ is the rate class i was expected to achieve in the same period; the latter is given by

$$target_r_i[n] = \frac{w_i[n]}{\sum_j w_j[n]} \cdot C_{cck11},$$

where $w_i[n]$ is the weight of class i and C_{cck11} is the maximum throughput that can be achieved by 802.11 when complementary code keying at 11 Mbps is used; due to the physical and MAC layer overheads, this maximum throughput cannot be larger than 6 Mbps. The variable $D_i[n]$ in (1) will be negative if node i has lost service, and positive if node i has received excess service up to period n .

The weight of class i will be updated at the end of each period according to

$$w_i[n+1] = w_{i,initial} + f_i[n], \quad (2)$$

where $f_i[n]$ is a function of $D_i[n]$. Function $f_i[n]$ must be bounded so as to avoid an excess increase of a class's weight, which can cause starvation of other classes. It should also result in a smooth decrease or increase of a class's weight. To achieve the above we propose the following function that has shown good performance in our experiments:

$$f_i[n] = \begin{cases} \ln\left(\frac{|D_i[n]|}{C_{cck11}} + 1\right), & \text{if } -w_{i,init} \cdot S_l \leq D_i[n] \leq 0 \\ -\ln\left(\frac{|D_i[n]|}{C_{cck11}} + 1\right), & \text{if } 0 < D_i[n] \leq w_{i,init} \cdot S_g \end{cases} \quad (3)$$

where S_l and S_g are, respectively, the maximum aggregate loss and gain we will account for in our WLAN; $w_{i,init}$ is the initial weight of class i . Values of $D_i[n]$ larger than $w_{i,init} \cdot S_g$ or smaller than $-w_{i,init} \cdot S_l$ aren't accounted for. The value of S_l is determined by considering the available resources so that our algorithm can indeed compensate the amount of losses it accounts for. The value of S_g should be such that a class that has received excess service in the past won't starve or suffer weight reduction for a prolonged period. Furthermore, the maximum amount of loss or gain of a class in any period is proportional to its initial weight, as is the compensation or service reduction it will receive.

C. Performance Improvement Algorithm

In order to address the unfairness and reduced throughput in WLANs that contain nodes with transmission rate lower than 11 Mbps, weight adjustment can take into account each node's channel occupation time for transmitting one packet. This can be achieved by extending (2) to

$$w_i[n+1] = (w_{i,initial} + f_i[n]) \cdot l_i[n] \quad (4)$$

where $f_i[n]$ is computed from (3), $w_{i,init}$ is the initial weight of class i , and l_i is the transmission rate coefficient which

is the inverse of the channel occupation time needed for the transmission of one packet. Table I shows the transmission rate coefficient for different transmission rates of 802.11b. These values can be calculated analytically considering the MAC and physical layers overheads, and were verified on our testbed; they can be considered good approximations when the packet size is larger (above 150 bytes) than the 802.11b frame overhead (52 bytes).

TABLE I
TRANSMISSION RATE COEFFICIENT.

Modulation (802.11b)	Transmission Rate (Mbps)	Channel Occupation Time (time units)	Tx rate Coefficient (l)
DBPSK (1 Mbps)	1	6	1/6
DQPSK (2 Mbps)	2	6/2	2/6
CCK-5.5 Mbps	4	6/4	4/6
CCK-11 Mbps	6	1	1

IV. IMPLEMENTATION AND TESTBED EVALUATION

The proposed dynamic CBWFQ scheme has been implemented on a Linux Box equipped with a Prism 2.5 wireless card driven by HostAP. The CBWFQ mechanism was implemented with the hierarchical token bucket queuing discipline (HTB) [6]. A set of parsers gather the necessary throughput and transmission rate statistics from the Linux `tc` tool [7] and the HostAP driver, respectively.

The experimental evaluation of the proposed scheme used the network topology shown in Fig. 2. The host on the wired part of the testbed had an ftp server installed that acted as a background traffic sink, and a streaming server (the Darwin streaming server) that was transmitting unicast streams containing an mpeg-4 encoded movie file. The testbed was placed in a typical office environment, where signal fading occurred as people moved around. Furthermore, all ACK and DATA classes were initially assigned equal weights (16,67%). Parameters S_l and S_g were set to 25 Mbits for all experiments, so that each node's ACK and DATA classes would be compensated for losses up to 4.16 Mbps. Compensating this amount of losses not only improves fairness but also showed to enhance the stability of the throughput compensation algorithm.

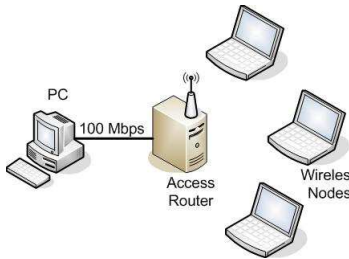


Fig. 2. Topology of the experimental testbed

Fair bandwidth distribution in case of bursty errors. The target of this experiment is to demonstrate the effectiveness of the throughput compensation algorithm in redistributing bandwidth fairly when location-dependent channel errors occur. Each of the three wireless nodes of Fig. 2 was receiving an ftp flow. Fig. 3 presents the throughput achieved by each node, over time, without using the dynamic CBWFQ mechanism. The results clearly show an unfair distribution of bandwidth

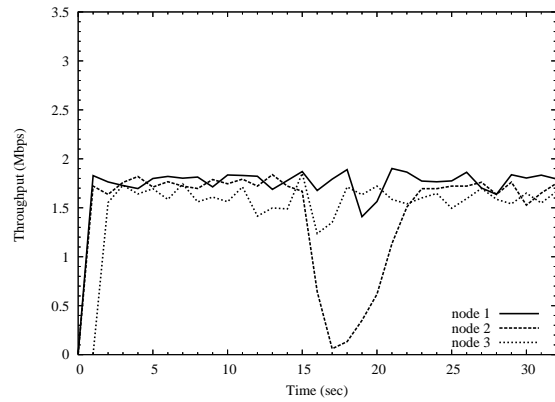


Fig. 3. Unfairness due to bursty channel errors.

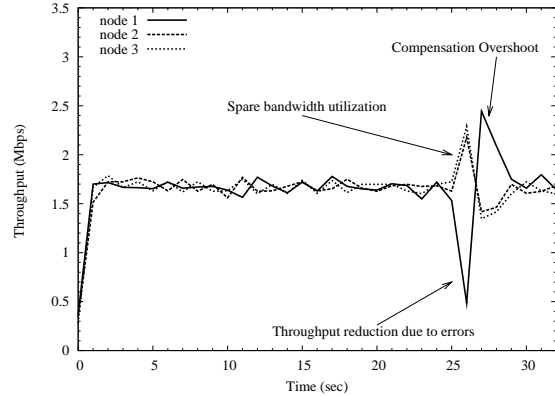


Fig. 4. The dynamic CBWFQ scheme achieves fairness in the presence of bursty errors.

between the three nodes. In particular, node 2 suffered severe losses during the experiment and achieved lower overall throughput compared to the other two nodes. This was due to location-dependent channel errors that occurred when people walked between the laptop and the access router. Fig. 4, on the other hand, presents the throughput achieved by each ftp flow, when the same experimental setup was used but with the dynamic CBWFQ mechanism activated. In this case, although node 1 lost service at the 26th second, it was compensated for its lost service as indicated by the throughput overshoot that followed each loss. Furthermore, it can be seen that when node 1 failed to utilize its assigned throughput, the other nodes utilized the spare resources thus increasing the aggregate channel utilization.

Uplink-Downlink Unfairness. This experiment demonstrates the effectiveness of our mechanism in an uplink-downlink unfairness scenario. In this experiment nodes 1 and 2 were receiving a unicast video stream while node 3 generated three uplink ftp flows towards the wired host. We used three uplink ftp flows so that there would still be uplink traffic in case one or two of the flows decreased their congestion window simultaneously due to data packet losses. Furthermore, we reduced the output buffer of the wireless router to 50 packets from 100 packets, which is its default value. This experimental setup caused the output buffer of the wireless interface at the access router to overflow and thus exhibit the uplink-downlink unfairness phenomenon. Fig. 5 shows that the uplink flow

achieves higher throughput compared to each video stream. Furthermore, over some periods the throughput of the uplink flow is higher than the aggregate throughput of both video streams. Fig. 6 presents the throughput achieved by each host in the same scenario but with our mechanism activated. The figure shows that by shaping the transmission rate of acknowledgment packets and scheduling them through their own queues we managed to control the uplink ftp flows' throughput and avoid overflows of the output buffer at the wireless interface due to acknowledgment and data packets cluttering in the same buffer.

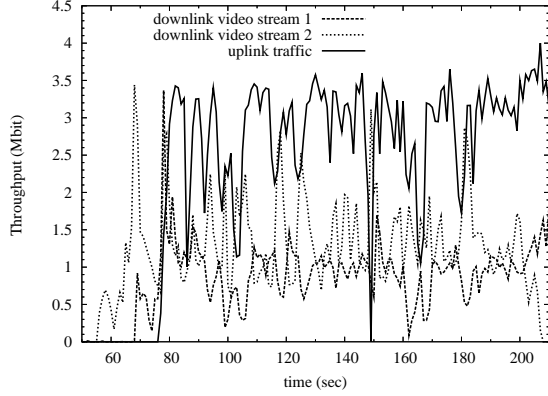


Fig. 5. Uplink-downlink unfairness scenario.

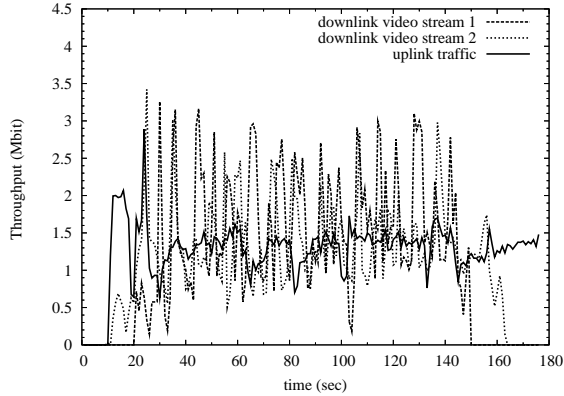


Fig. 6. The dynamic CBWFQ scheme's ability to control resource sharing enables us to solve uplink-downlink unfairness.

Performance anomaly of 802.11. This experiment illustrates the impact of the performance anomaly problem on voice-over-IP streams and the way our mechanism tackles with this issue. The experiment incorporated a bidirectional VoIP call between nodes 1 and 2, and three uplink ftp flows generated by node 3, which was manually configured to transmit at 1 Mbps. The VoIP call was generated using gnomemeeting with the G.711 codec and no silence detection. This experiment was conducted twice, once without our mechanism and once with the mechanism activated. In Fig. 7 we present the interarrival jitter for the voice packets that were transmitted by node 1 to node 2. The interarrival jitter J was calculated for each data packet i received by node 2, using a weighted average of the delay jitter between consecutive packets. Hence,

$$J = J + (|D(i-1, i)| - J)/16,$$

where $D(i-1, i)$ is the delay jitter of consecutive packets

$$D(i-1, i) = (R_{i-1} - R_i) - (S_{i-1} - S_i),$$

where R_i is the time of arrival of packet i at node 2 and S_i is the departure time of the same packet from node 1. In case of increased interarrival jitter the users participating in the VoIP call experience periods of good voice quality followed by periods of low voice quality. The lower graph in Fig. 7 shows that the dynamic CBWFQ mechanism in a performance anomaly scenario can significantly decrease the interarrival jitter experienced by VoIP packet streams; this is achieved by reducing the weight of nodes with a low transmission rate.

Experiments illustrating the effect of the performance anomaly phenomenon on the aggregate throughput in 802.11 WLANs as well as the capability of the dynamic CBWFQ mechanism to resolve this issue were presented in [1].

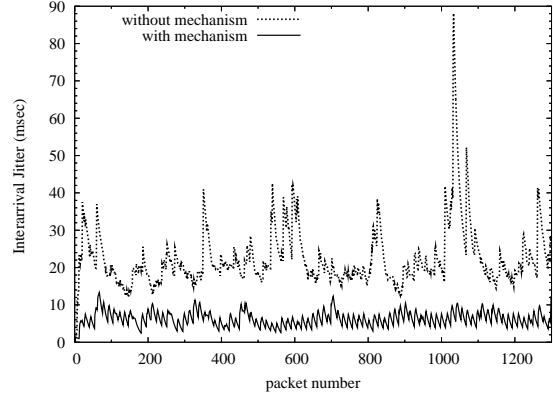


Fig. 7. Interarrival jitter for a VoIP call

Response time measurements for web traffic. This experiment points out the performance improvement that results from compensating service loss due to absence. Since this aspect focuses on bursty data traffic, Web browsing is used as a case study. An Apache web server was installed on the wired host and three objects of sizes 50, 152, and 500 KBytes were available for download. Two wireless stations were used for this experiment; the first one made an ftp transfer of a large file while the second one performed HTTP requests using the `wget()` utility. An exponentially distributed think time, with a mean of 20 seconds, was assumed. The wireless host that performed the HTTP requests was a Linux Box with the same setup as the Access Router. Each DATA class was initially assigned half of the available bandwidth. The experiment was comprised of 20 HTTP requests for each object and the mean response times is shown in Fig. 8. The performance improvement is significant for small objects, and smaller for large objects. This is due to the limitation imposed to the amount of service that a class can be compensated for, which is controlled by the S_l parameter.

V. RELATED WORK

Service differentiation in wireless networks has been studied by several researchers. In [8] the authors combine the class-based queuing mechanism with a channel state dependent packet scheduler and a compensation algorithm in a manner

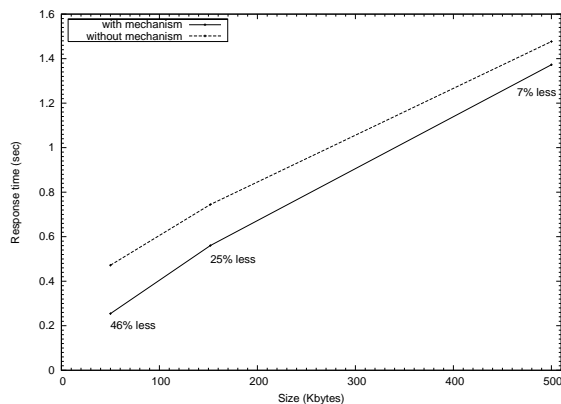


Fig. 8. Improving response time for Web traffic.

that is conceptually similar to ours. However, they don't address the problem of uplink traffic control and there is no accounting for losses due to absence. Furthermore, they propose the design of a special purpose scheduler, whereas our algorithms are independent modules communicating with the scheduler only through the class weight parameter. Their algorithm prohibits the transmission of a class that has received excess service in the presence of a class that has suffered losses. Hence there is no provision for avoiding starvation. Finally, the proposed link state estimation algorithm proposed, is based on RTS/CTS messages and alteration of the maximum retransmission attempts.

With regard to fairness, of particular interest are the works in [9], [10] and [11] presenting fair scheduling algorithms. In [9], Zhu *et al.* introduce the notion of accounting absence as loss of service but they don't associate it with weight adjustment as a way to improve fairness and performance. Furthermore, deployment of all the above algorithms in 802.11 networks requires the construction of a specialized scheduler, significant changes at the MAC layer and modifications of the end systems.

The wireless link-state estimation problem has also been addressed by many researchers. In [12], Aida *et al.* make use of the mean SNR in order to characterize the state of the wireless links. The major drawback of this method is the lack of knowledge of the SNR value at the region of the mobile host, which must be transmitted explicitly to the access point. Another line of research [13] suggests to employ MAC level acknowledgment failures as a criterion for link quality and change the retransmission attempts accordingly. These algorithms, however, require changes at the MAC layer and modifications of the end systems, whereas our approach utilizes the transmission rate as an indication of a link's quality, this information is readily available at the Access Router.

Recently, in [2], Pilosof *et al.*, proposed the dynamic adjustment of the TCP window in order to control uplink traffic. This method, however, could not coexist with security protocols, e.g IPsec, that encrypt a packets content. It also requires exact knowledge of the active TCP flows number in order to share bandwidth fairly, which is difficult to obtain in the presence of idle TCP flows. In contrast to this method,

our approach for controlling uplink traffic requires only the number of hosts that are associated with the Access Router.

In [4] the authors redefine fairness in terms of time shares. By granting temporal fair share of the channel to each traffic source they improve fairness and performance even in multi-rate capable physical layers. However, their algorithm does not support service differentiation.

Finally, it is interesting to contrast the dynamic CBWFQ mechanism proposed in this paper with the IEEE 802.11e standard supplement, which provides MAC layer functions for service differentiation. Unlike our mechanism, 802.11e can support only static service differentiation and there is neither compensation for losses nor anticipation for throughput degradation due to the multirate physical layer. Moreover, our dynamic CBWFQ mechanism, implemented at the network layer, is compatible with all existing 802.11 networks.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we addressed the problem of supporting service differentiation in existing WLANs based on IEEE 802.11. In such networks the direct employment of service differentiation algorithms developed for wired networks is not effective, due to the peculiarities of the wireless channel. However, utilization of existing mechanisms, properly supported with algorithms that exploit cross-layer information to deal with fairness and performance issues, proved to be an efficient and practical solution in real life scenarios.

Future work includes implementing the dynamic CBWFQ scheme on top of a WMM (WiFi Multimedia) capable wireless driver, and on a control station that is connected with more than one legacy access points.

REFERENCES

- [1] V. Siris and G. Stamatakis, "A dynamic cbwfq scheme for service differentiation in 802.11 w lans," *PIMRC'05*, September 2005.
- [2] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha, "Understanding tcp fairness over wireless lan," *IEEE INFOCOM*, 2003.
- [3] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," *IEEE INFOCOM 2003*, 2003.
- [4] Y. Yuan, D. Gu, W. Arbaugh, and J. Zhang, "Achieving packet-level quality of service through scheduling in multirate w lans," *VTC IEEE*, 2004.
- [5] V. Gambiroza, B. Sadeghi, and E. Knightly, "End-to-end performance and fairness in multihop wireless backhaul networks," *MobiCom'04*, Oct. 2004.
- [6] <http://luxic.cdi.cz/~devik/qos/htb>, "Htb home."
- [7] [http://www.rns-nis.co.yu/~mps/linux tc.html](http://www.rns-nis.co.yu/~mps/linux%20tc.html), "Linux qos control tool."
- [8] C. Fragouli, M. Srivastava, and V. Sivaraman, "Controlled multimedia wireless link sharing via enhanced class-based queueing with channel state dependent packet scheduling," *IEEE INFOCOM'98*, March 1998.
- [9] H. Zhu and G. Cao, "On improving service differentiation under bursty data traffic in wireless networks."
- [10] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *IEEE/ACM Transactions on networking*, vol. 7, August 1999.
- [11] T. Ng, I. Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location dependent errors," *IEEE INFOCOM'98*, March 1998.
- [12] H. Aida, Y. Tobe, M. Saito, and H. Tokuda, "A software approach to channel-state dependent scheduling for wireless lans," *WOWMOM*, 2001.
- [13] P. Bhagwat, P. Bhattacharya, A. Krishma, and S. Tripathi, "Enhancing throughput over wireless lans using channel state dependent packet scheduling," *IEEE INFOCOM'97*, April 1996.