Smart Traffic Scheduling in 802.11 WLANs with Access Point

M. Bottigliengo, C. Casetti, C.-F. Chiasserini, M. Meo
CERCOM - Dipartimento di Elettronica
Politecnico di Torino
Torino, Italy

***Abstract*— In distributed systems like 802.11 WLANs, simple requirements like providing fair access to all users are hard to get by because of the random access protocol and the unpredictability of the wireless channel. The main contribution of this paper is the proposal of an LLC-layer algorithm that is implemented at both AP and WSs. The algorithm aims at guaranteeing fair access to the medium to every user, by awarding longer transmission opportunities to WSs that experienced short channel failures. At the same time, an adaptive contention window setting protects downlink traffic from the unbalancement typical of WLANs with AP. We outline the proposed solution and present a simulation study that shows the effectiveness of the new algorithm in comparison to the standard 802.11b implementation both under a TCP and a UDP traffic scenario.**

I. INTRODUCTION

Wireless local area networks (WLANs) based on the IEEE 802.11b technology are experiencing a huge popularity and widespread deployment. Currently, the most successful 802.11b network configuration includes an access point (AP) and several wireless stations (WSs); the AP provides connectivity with the wired part of the network thus allowing Internet services to be extended to wireless users. On the one hand, a great effort has been done to support quality of service (QoS) provisioning in *wired* networks. For instance, at the network layer, the Differentiated Services (DiffServ) architecture [1] was introduced to provide aggregate users with different levels of stochastic QoS. On the other hand, one of the major deficiencies in the current 802.11b standard is the lack of QoS support. Among the several limitations of 802.11b WLANs in providing QoS, in this work we address the following: (i) the location-dependent capacity and errors over the wireless channel [1]; (ii) the uplink/downlink unfair channel access in WLANs with AP [2], [3].

The goal of our paper is to improve QoS support in 802.11b WLANs by designing a scheduling scheme that compensates for channel errors and balances transmission opportunities among all flows. Our contribution is twofold: (i) providing flows belonging to the same QoS class with fair access to radio resources; (ii) equalizing channel access between uplink and downlink flows. This is achieved through the design of a channel-aware scheduling scheme along with an adaptive Contention Window (CW) setting at the AP as first proposed in [4].

II. RELATED WORK AND MOTIVATIONS

Fairness in wireless networks has been studied under various network scenarios, ranging from cellular networks and WLANs to ad hoc networks.

With regards to WLANs, of particular interest are the works in [5], [6], [7], [8], [9], presenting new, distributed, MAC algorithms aimed at providing fair channel access. Also relevant to our work are some studies on MAC short-term fairness and its impact on the TCP protocol. In [2] the authors observe a significant unfairness between uplink and downlink flows when the DCF is employed in a 802.11 WLAN with AP. The reason is that in a WLAN with N stations there are N uplink CSMA instances contending with only one downlink CSMA instance (the one at the AP). Unfairness between uplink and downlink TCP flows in 802.11 WLANs is observed in [3], but different causes are identified. There, the authors blame the unfairness on buffer size availability at the AP and propose a solution that is based on TCP receiver window manipulation at the AP.

In our study, we observe the cause of unfairness outlined in [2], in UDP flows, but only marginally in TCP flows due to their closed-loop nature. Conversely, the unfairness identified in [3] is observed in TCP flows and not in UDP flows. Our proposal introduces an integrated approach that aims at providing fairness to uplink and downlink flows regardless of the transport protocol used.

III. REFERENCE SCENARIO

In this section, we first describe the network and traffic assumptions we made in order to develop our algorithm;

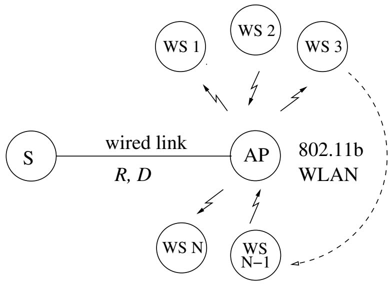

Fig. 1. Network scenario

we then introduce the channel model.

A. Network and Traffic

We consider a wireless-cum-wired network scenario as shown in Figure 1. A fixed node S is connected with an AP through a wired link of speed R and propagation delay D. This link is over-provisioned so that no packets are dropped at its ends. The wireless portion is an 802.11b WLAN with N WSs. The RTS/CTS (Ready-To-Send/Clear-To-Send) mechanism is employed. Also, we assume that WSs and the AP operate at a data rate of 11 Mbps, even though the 802.11b standard provides for other rates. We deliberately chose not to have stations switch to slower rates upon channel failures in order to avoid compromising the WLAN overall performance, as described in [10].

At transport layer, TCP connections and UDP flows are established between each WS and the fixed node S. In particular, given N wireless stations, $N/2$ traffic relations are established in the uplink direction (from a WS to S), while the remaining $N/2$ are in the downlink direction.

B. Channel Model

Modeling the wireless channel behavior is very challenging due to the burstiness and time correlation of wireless errors. Furthermore, WLANs may experience location-dependent channel errors, in that a WS can correctly communicate with the AP, while at the same time another one may suffer frame drops due to errors on the channel. In order to consider all these aspects, an independent error model for each communicating pair of nodes was introduced.

An error model is represented by a three-state discrete-time Markov chain. The Markov chain time slot is equal to the *aSlotTime* parameter of the 802.11b MAC sublayer, i.e., $20\mu s$. Errors over the channel occur in the states *long bad (LB)* and *short bad (SB)*, while the *good (G)* state is error-free. Thus, a frame transmission is successful only if the error model is in state *G* for all slots it takes to the frame to be transmitted, while it fails otherwise. The difference between the long bad LB and short bad SB states is the time correlation of errors: LB corresponds to long bursts of errors, *SB* to short ones. The probability that the Markov chain moves to the LB state given that it leaves the state G, i.e., the probability that an error burst is long is set to 0.05. We assume that the average number of bad slots experienced when the states LB and SB are entered, are respectively equal to 65160 (1.3 s) and 2000 (0.04 s). The average number of consecutive error-free slots is set to 50000 (1 s).

IV. An Integrated Approach to Fairness

As stated above, we set as our aims the provision of fairness to both TCP (closed-loop) and UDP (open-loop) flows. In addition, we want to achieve fair channel access by both uplink and downlink flows. To this end, we introduce in this section an integrated solution to fairness provision that is hinged upon a channel-aware scheduling scheme and an adaptive contention window setting.

A. Channel-aware Scheduling Algorithm

The goal of our scheduling algorithm is to improve the fairness among wireless stations that may experience location-dependent channel capacity and errors.

To achieve these results, we proceed as follows. At the AP LLC layer, we introduce a separate queue for each WS associated to the AP, while only one queue is implemented at the WS LLC layer. A channel condition estimator is associated to each queue, and transmission is allowed only for those queues whose channel is estimated to be *good*, i.e., such that 11 Mbps speed can be attained. Queues attempting to access the wireless medium with a *bad* channel will be allowed to transmit again when their channel becomes *good*.

The building blocks of our algorithm are the following:

- channel state estimation
- queue selection and service

The algorithm is implemented both at the AP and at WSs. Note that, since only one queue is used at each WS, the queue selection in not needed at WSs. Below, we briefly describe these building blocks for the most general case, i.e., for the AP.

1) Channel State Estimation: As mentioned above, the AP estimates the channel conditions for each contending WS. Thus, a flag is associated to each LLC queue at the AP, indicating the corresponding channel state. The flag can take three values: GOOD, BAD or PROBE.

GOOD: The AP sets the flag to GOOD whenever it receives, from the corresponding WS: (i) a MAC-layer acknowledgment in response to a data frame, (ii) a CTS frame in response to an RTS frame, or (iii) an error-free data frame or RTS.

BAD: The AP sets the flag to BAD after a transmission failure. To tell if the cause of a transmission failure is due to collisions or channel errors, we resort to a careful selection of the values for the Short Retry Limit (SRL) and the Long Retry Limit (LRL). These parameters control the number of transmission attempts without receiving an acknowledgement; the number of attempts are tracked by a Long Retry Counter (LRC) and a Short Retry Counter (SRC). In all likelihood, the LRC is incremented when the transmission of a frame longer than the RTS threshold fails due to channel errors [8]. Instead, the SRC is incremented both because of channel errors and because of collisions over the shared medium. When LRC (SRC) reaches the LRL (SRL) value, the MAC layer abandons the transmission of the frame and it signals the failure to the LLC layer. While LRC increments allow a detection of channel errors, the interpretation of SRC increments is ambiguous. We then apply an empirical method: we performed some simulations with a large number of contending stations (namely, 40) and ideal channel conditions; we found that the probability of SRC ever reaching values greater than 4 was almost negligible (i.e., of the order of 10^{-4}). We can therefore assume that it is highly likely that values of SRC larger than 4 are due to channel errors. Consequently, we set SRL to 4 and LRL to 0. Upon reaching these values, the AP sets the flag to BAD and the frame discarded by the MAC is maintained at LLC layer and used to probe the channel afterwards.

PROBE: The AP switches the flag from BAD to PROBE when a configurable timeout, that we named PTIMER, expires. PTIMER starts to run whenever the channel state switches to BAD, and its initial value is doubled when a transition from PROBE to BAD occurs. The duration of PTIMER is reset to its initial value upon a transition from PROBE to GOOD. A WS whose queue flag has a PROBE value can transmit a frame to check the new channel state.

2) Queue Selection and Service: The queue selection algorithm is sketched in Figure 2 and it is executed every time the MAC layer requests a new frame to send. At the LLC layer, queues are served in a round-robin fashion. Each queue has associated with it a counter of upcoming transmission opportunities so far accumulated (in the following, we indicate the upcoming transmission opportunities as “Tokens”). The number of Tokens is incremented by 3 the first time each queue is visited

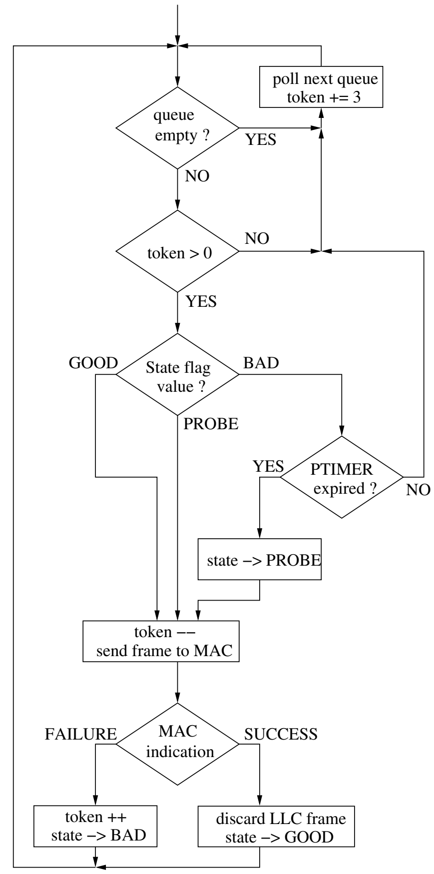

Fig. 2. Queue service (top) and queue selection (bottom) flow charts

in a round. We found in our simulations that this increment is high enough to quickly recover from temporary transmission blackouts, while it limits the frame queue latency.

Initially, the algorithm checks if the currently polled queue is empty or if it has spent all its Tokens: if at least one of these two conditions is true, the next queue is polled. Otherwise, the value of the associated channel flag is evaluated. In case of GOOD or PROBE channel values, the queue is served. When the channel is BAD, the PTIMER is inspected: if still running, the next queue is polled; otherwise, the flag is set to PROBE, and the queue is served.

Before sending a frame to the MAC Layer, the number of corresponding Tokens is decremented to consider the transmission opportunity that the queue will spend. From now on, the process of queue selection relies on indications from the MAC layer regarding the success or the failure of the frame transmission. In the case of successful transmission, the corresponding LLC frame can be discarded and the channel state is set to GOOD (this switching occurs only if the previous state was PROBE). Otherwise, if the transmission fails, the channel flag is set to BAD and the Tokens are incremented since the corresponding queue has not consumed one of them. In this case the LLC frame is not discarded and it will be used to probe the channel afterwards.

We also remark that each frame has an internal timestamp associated to it: frames that have been waiting in the queue for a time greater than a STALE_TIMEOUT value are discarded. We introduced this timeout to avoid an excessive latency during long channel error periods. We found by simulation that setting STALE_TIMEOUT to 1 s allows the average end-to-end delay to remain below reasonable values.

B. Contention Window Adaptation

The algorithm described above can achieve good fairness among downlink flows, but it has no capability to equally share the available bandwidth between uplink and downlink streams. This disparity is due to the limited transmission opportunities gained by the AP, if compared with WSs. Indeed, under the same MAC parameters, the probability to gain access to the channel is the same for the AP and all the N WSs, but the AP has N LLC queues to serve. Consequently, unless higher-layer, closed-loop mechanisms (like TCP) regulate the traffic, uplink flows will obtain a better throughput than downlink ones, since each WS only has one LLC queue.

To overcome the problem, we try to differentiate the minimum CWs (CW_{min}s) between AP and WSs. There are two possibility: (i) decreasing the AP CW_{min} (denoted by CW_{min_AP}), or (ii) increasing the WSs CW_{min}. Apparently the former solution seems to be the best (since only a parameter must be changed), but it has a drawback: with a large number of WSs, the CW_{min_AP} becomes too small. Therefore we adopt the second solution: given N WSs, their CW_{min} are set to $(N \times (CW_{min\_AP} + 1)) - 1$, where CW_{min_AP} is set to 31. Notice that the maximum CW is set to 1023 at the AP, while it is set to 4095 at the WSs.

The proposed scheme assumes that each WS MAC layer is aware of the value of N and CW_{min_AP}. The former can be easily computed by the MAC from the unique addresses observed during a pre-established time period. The second is a parameter that the AP normally broadcasts to all WSs during the Beacon period.

V. Simulation Scenario

Simulations, run under the ns-2 simulator, use the network scenario depicted in Figure 1. The wired link between the AP and the fixed node S has a 55 Mbps bandwidth and its propagation delay is equal to 2ms. We assume that there are no hidden stations.

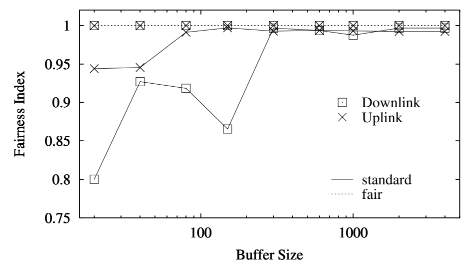

Fig. 3. Fairness index of uplink and downlink TCP connections as a function of buffer size at the AP (20 WSs)

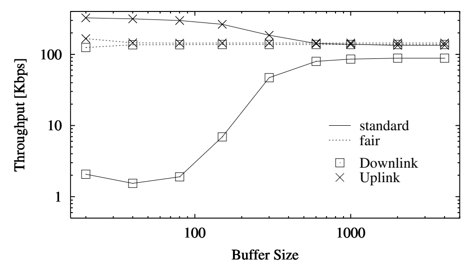

Fig. 4. Throughput of uplink and downlink TCP connections as a function of buffer size at the AP (20 WSs)

TCP sources generate greedy traffic, while UDP flows offer an overall traffic equal to 2 Mbps; therefore the transmission rate of a single UDP source decreases when the number of WSs increases. To avoid synchronization effect during simulations, packet departure times are randomly staggered by a very small interval. The TCP version used in our simulation is NewReno. The Maximum Segment Size (MSS) of TCP segments is equal to 1000 bytes. The initial congestion window and its maximum value are set, respectively, at 1 and 100 segments. For UDP flows, the maximum packet size is equal to 250 bytes.

SRL and LRL are set to 7 and 4 when the standard 802.11b MAC is simulated, while they are set to 4 and 0 when our scheme is used (as explained in Section IV). We set the RTS threshold at 400 bytes so that TCP ACKs and UDP segments are never transmitted using the RTS/CTS handshaking. LLC queues at WSs are 20 data frames long, while at the AP we vary this parameter between 20 and 4000.

VI. Numerical Results

The first set of results show the fairness and throughput achieved by uplink and downlink TCP connections in a standard WLAN (tagged "standard") and in one using our integrated fairness approach (tagged "fair"). Figure 3 reports Jain's fairness index [11] computed on throughputs of uplink or downlink TCP connections as a function of the AP buffer size for the case of 20

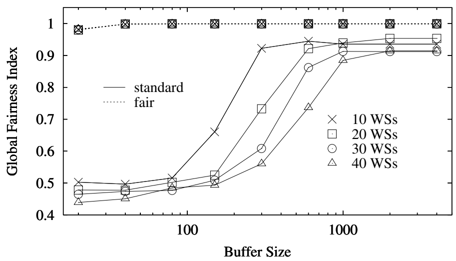

Fig. 5. Global fairness of TCP connections as a function of buffer size at the AP

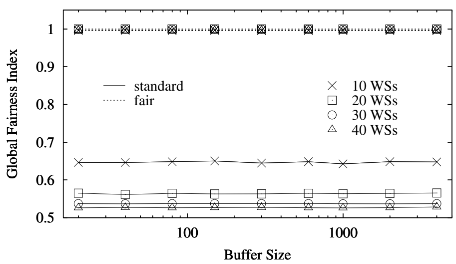

Fig. 6. Global fairness of UDP flows as a function of buffer size at the AP

WSs. It can be seen that the unfairness in standard WLAN is only mitigated by larger buffers, that allow the closed-loop control by TCP to kick in. Our approach is consistently fair no matter what buffer size is used. Similarly, Figure 4 displays the throughput disparity between uplink and downlink in the standard case, and the equalization introduced by our approach. Very similar results were obtained varying the number of WSs, as is confirmed by Figure 5, where the global fairness index (i.e., including both uplink *and* downlink flows in the fairness computation) is shown.

The last two figures present results obtained by UDP flows: here, buffer size does not affect the performance of UDP flows, which retain a marked unfairness in the standard case, while our approach again ensures almost perfect fairness among all flows. Figure 6 presents the global fairness index of standard and fair case. Lastly, Figure 7 details the case of 20 WSs, highlighting the superior channel utilization achieved by our scheme, along with the fair sharing between uplink and downlink flows.

VII. Conclusions

We have presented an integrated fairness approach for 802.11 WLANs with AP that combines fairness provisioning to flows belonging to the same QoS class and to uplink/downlink flows. Simulation results have shown a marked improvement in terms of fairness achieved

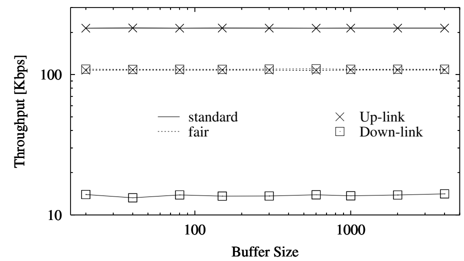

Fig. 7. Throughput of UDP flows as a function of buffer size at the AP (20 WSs)

by our scheme in comparison with standard 802.11 WLANs.

References

[1] S. Lu, V. Bharghavan, and R. Srikant, “Fair scheduling in wireless packet networks,” *ACM Sigcomm’97,* Cannes, France, Sep. 1997.

[2] A. Grillo and M. Nunes, “Performance evaluation of IEEE 802.11e,” *IEEE PIMRC’02*, Lisboa, Portugal, Sep. 2002.

[3] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha, “Understanding TCP Fairness over Wireless LAN,” *IEEE Infocom 2003*, San Francisco, CA, USA, Mar. 2003.

[4] G. Bianchi, L. Fratta, and M. Oliveri, ”Performance Evaluation and Enhancement of the CSMA/CA MAC Protocol for 802.11 Wireless LANs,” *PIMRC’96*, Taipei, Taiwan, 1996.

[5] J. L. Sobrinho and A. S. Krishnakumar, “Real-time traffic over the IEEE 802.11 medium access control layer,” *IEEE Globecom’96,* Nov. 1996.

[6] D.-J. Deng and R.-S. Chang, “A priority scheme for IEEE 802.11 DCF access method,” *IEICE Transactions on Communications,* vol. E82-B, no. 1, pp. 96–102, Jan. 1999.

[7] M. Barry, A. T. Campbell, and A. Veres, “Distributed control algorithms for service differentiation in wireless packet networks,” *IEEE Infocom’01*, Anchorage, AK, USA, Apr. 2001.

[8] N. H. Vaidya, P. Bahl, and S. Gupta, “Distributed fair scheduling in a wireless LAN,” *IEEE/ACM MobiCom’00,* Boston, MA, USA, Aug. 2000.

[9] I. Aad and C. Castelluccia, “Differentiation mechanisms for IEEE 802.11,” *IEEE Infocom’01,* Anchorage, AK, USA, Apr. 2001.

[10] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, “Performance anomaly of 802.11b,” *IEEE Infocom’03*, San Francisco, USA, Mar. 2003.

[11] R. Jain et al., “A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems”, DEC RR TR-301, Sep. 1984.

0-7803-7955-1/03/$17.00 (C) 2003 IEEE