

Short-term Fairness for TCP Flows in 802.11b WLANs

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

Abstract—Wireless Local Area Networks (WLANs) based on the IEEE 802.11 technology are becoming increasingly popular and widely deployed. However, the growing need for Quality of Service (QoS) guarantees is difficult to implement in distributed systems like WLANs, where the random access protocol and the unpredictability of the wireless channel make it difficult their interaction with well-established architectures like DiffServ. Even without trying to provide deterministic QoS guarantees, simpler requirements are hard to get by. For example, the basic requirement of providing fair access to all users is conflicting with the nature of higher-layer protocols: TCP is fair only under certain conditions, hardly met by 802.11b WLANs. Another basic requirement is the protection for short-lived TCP flows, that are sensitive to losses during the early stages of the TCP window growth. The main contribution of this paper is the proposal of an LLC-layer algorithm that can be implemented on both AP and WSSs. The algorithm aims at guaranteeing fair access to the medium to every user, by awarding longer transmission opportunities to WSSs that experienced short channel failures. At the same time, such award mechanism can protect short-lived flows while they strive to get past the critical "small window regime". 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.

Keywords: System design, Simulations.

I. INTRODUCTION

In the short span of four years since becoming a standard, IEEE 802.11b Wireless Local Area Networks (WLANs) look set to revolutionize the way people are perceiving access networks. By becoming a ubiquitous facility, available both in public areas such as airports, cafes, museums, workplaces, and in private homes, it is conceivable that people will soon expect that the same (or a comparable) QoS they receive on wired networks is also available to wireless networks. To wit, one could argue that saying "wired networks" and "QoS" in the same breath has often been a debatable issue. The

provision of QoS guarantees has often been hindered by several factors including, but not limited to, the inherently best-effort service model of the IP protocol, the uncertainty in characterizing the traffic generation process, the unavailability of QoS routing algorithms and protocols that could be deployed on a global basis, and the poor scalability of several attempts at defining an articulate QoS paradigm. Among the several solutions proposed in the literature in the past years, the most successful one (both in terms of consensus and of actual deployment) has to be the DiffServ architecture [1]. DiffServ received the endorsement of IETF and routers today offer a wide range of support for DiffServ capabilities.

On top of these observations, wireless networks offer a whole set of new challenges to tackle. Traditionally, obstacles to QoS provisioning in wired LANs are: their distributed nature, the burstiness of traffic, the unpredictability of the number of users, the random access protocol. Beside these, WLANs using the 802.11b technology further add some of their own, such as poor channel quality depending on relative position of Wireless Stations (WSSs), interference from hidden terminals, and the anomaly observed in [2], due to the different speeds at which WSSs transmit. This lack of determinism has several drawbacks. Among them, the guarantees, albeit of a statistical nature, that architectures like DiffServ provide cannot be extended to users of wireless communities, because even the most privileged traffic flows (in wired parts of the network) can be unpredictably delayed or discarded by temporary channel failures.

Additional mechanisms are therefore necessary if QoS guarantees are to be provided. If an extension of DiffServ capabilities is to be engineered for wired-cum-wireless networks, two issues are prominent: differentiation of traffic classes (and its support in network elements) and fair access to network resources for all flows within the same traffic class.

In this paper, we address the latter issue, that is fairness among flows and, more specifically, among TCP

flows. We consider an 802.11b-based WLAN where WSSs access the network through an Access Point (AP). In such environment, fairness is compromised, among other things, by location-dependent channel capacity and errors [3]. On top of that, under TCP traffic scenarios unfairness is exacerbated by the different throughput and completion time achieved by long- and short-lived TCP flows, a problem that is closely related to the intuition of long- and short-term fairness¹. In particular, short-lived flows, representing the majority of today's Web traffic, suffer from packet losses occurring during or just past the three-way handshake phase, when the TCP congestion window size may not be large enough so as to trigger the Fast Recovery algorithm. Such flows are delayed by (possibly) repeated timeouts that are affecting them only because of their state, while longer flows manage to avoid timeouts thanks to the reception of duplicate ACKs that enable Fast Recovery. A similar situation occurs right after a timeout, when a flow is more vulnerable because of its small window, and the danger of repeated timeouts caused by erroneous packet marking is consistent. Therefore, the time horizon over which fairness is guaranteed could, if long-term fairness were chosen, result in unwanted penalties for short-lived flows. For the reasons outlined above, we focus on the provision of short-term fairness to TCP flows in 802.11b WLANs.

The main contribution of this paper is the proposal of an LLC-layer algorithm that can be implemented at both AP and WSSs. The algorithm aims at guaranteeing fair access to the medium to every user, by awarding longer transmission opportunities to WSSs that experienced short channel failures. The award mechanism works by monitoring the successful medium accesses over a measurement window, and allowing short or long bursts depending on the WSS's recent history. By its design, this mechanism also favors short-lived TCP flows, which, thanks to their small congestion window (and hence a recent history of low throughput), always have the upper hand when competing with longer-running flows.

The paper is structured as follows: Section II reviews recent contributions from the literature on the topic of fairness in wireless networks; an outline of the reference scenario in which our algorithm is supposed to operate is then presented in Section III; the description of the algorithm is in Section IV; the simulation settings and numerical results can be found in Sections V and VI,

¹A system is said to be long-term fair if all users gain equitable access to its resources in the long run, although there may be transient periods of unbalanced access; short-term fairness, instead, refers to equitable share of resources in the short run. Usually, short-term fairness implies long-term fairness, but not vice-versa [16].

respectively; Section VII concludes the paper.

II. RELATED WORK

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

An initial study on fairness in a packet cellular environment is presented in [3], where Lu *et al.* first address the issue of fairness among wireless users, in presence of location-dependent channel capacity and errors. They propose a centralized scheduling algorithm to be performed at the base station, which relies on a special TDMA-based MAC algorithm. Their solution is based on a modified weighted round-robin technique, where the base station maintains *credits* for lagging flows and *debits* for leading flows, in order to compensate lagging flows in future times. Other centralized solutions appear in [4], [5], [6], where the goal is still to improve fairness in presence of location-dependent errors over the wireless channel.

Several proposals addressing the problem of fairness in ad hoc networks have appeared. In particular, research on the interactions between TCP and the 802.11 MAC layer has been carried out in [7], [8], [9]. However, these solutions specifically address mobility and multi-hop communication issues, of critical importance in an ad hoc network environment.

With regards to WLANs, of particular interest are the works in [10], [11], [12], [13], [14], presenting new, distributed, MAC algorithms aimed at providing fair channel access. A scheduling scheme for real-time traffic, the so-called *blackburst*, is proposed in [10]. This solution uses channel jamming to determine the real-time source with the longest waiting time, i.e., the station with the highest priority. In [11], the authors take station priorities into account and suggest that the backoff period be changed according to a station priority: the lower the priority, the higher the maximum backoff period for the station. A similar approach is adopted in [12], where two distinct backoff periods are used for two priority classes. In [13] the authors propose a MAC algorithm derived from the 802.11 DCF. Their scheme computes the backoff period for the stations so that the obtained bandwidth share closely matches the self-clocked fair queueing scheme. The work in [14] suggests three mechanisms for service differentiation in the IEEE 802.11-based WLANs. The authors propose to scale the contention window, vary the inter-frame spacings and change the maximum frame length for the stations, depending on the priority level of the station flow. The performance of the differentiation mechanisms is assessed under both a TCP and a UDP traffic scenario.

The solution that we present differs from all above proposals in that it operates at the link layer leaving the 802.11 MAC layer mostly unchanged.

Also relevant to our work are some studies on MAC short-term fairness and its impact on the TCP protocol. In [15] 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). In our study, we do not observe this cause of unfairness since we deal with TCP, i.e., a closed-loop protocol, as will be explained in Section VI. Unfairness between uplink and downlink TCP flows is also observed in [17], but different causes are identified. There, the authors study the interaction between TCP and the 802.11 MAC protocol in presence of mobile senders and receivers. They identify the cause of unfairness in the buffer size availability at the AP and propose a solution that is based on TCP receiver window manipulation at the AP. We do not tackle this aspect in our study, but restrict our analysis to unfairness caused by channel unavailability.

III. REFERENCE SCENARIO

In this section, we first describe the network and traffic assumptions we made in order to develop our algorithm; 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 overprovisioned so that no packets are dropped at its ends. The wireless portion is an 802.11b WLAN with N WSs. Each WS can directly communicate only with the AP, since we focus on AP-coordinated wireless networks, disregarding the ad hoc mode. 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 [2].

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

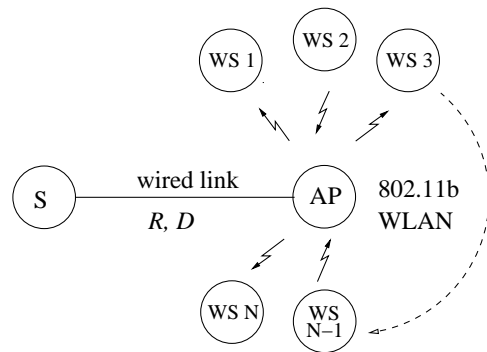


Fig. 1. Network scenario

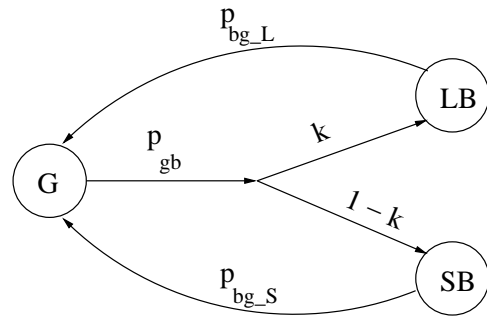


Fig. 2. Three-state error model

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. Therefore, given N WSs, we have N independent error models.

An error model is represented by a three-state discrete-time Markov chain, as the one shown in Figure 2. The Markov chain time slot is equal to the $aSlotTime$ parameter of the 802.11b MAC sublayer, i.e., $20\mu s$ in our simulations. 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 average number of bad slots experienced when the states LB and SB are entered, are respectively given by:

$$L_{LB} = \frac{1}{p_{bg_L}} \quad \text{and} \quad L_{SB} = \frac{1}{p_{bg_S}} \quad (1)$$

TABLE I
ERROR MODEL PARAMETERS

$L_G = 1/p_{gb} = 65160$ slots (1.3s)
$L_{SB} = 1/p_{bg-S} = 2000$ slots (0.04s)
$L_{LB} = 1/p_{bg-L} = 50000$ slots (1s)
$k=0.05$

where p_{bg-L} (p_{bg-S}) is the transition probability from state LB (SB) to G . Similarly, the average number of consecutive error-free slots is given by

$$L_G = \frac{1}{p_{gb}} \quad (2)$$

where p_{gb} is the probability to leave the state G . The parameter k is the probability that the Markov chain moves to the LB state given that it leaves the state G ; k also represents the probability that an error burst is long, or, in other terms, the fraction of long bursts over the total number of error bursts. The characteristics of the error model, such as the slot error probability and the frame error probability, can be easily derived from the steady-state behavior of the Markov chain.

Table I reports the error model parameter values that were used to derive the numerical results presented in Section VI.

IV. SCHEDULING ALGORITHM DESCRIPTION

The goal of our scheduling algorithm is twofold: first, to improve the fairness among wireless stations that may experience location-dependent channel capacity and errors; and, secondarily, to provide short-term fairness to TCP traffic in order to enhance the performance of short-lived flows.

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*. Upon switching to *good* channel state, a queue that has just experienced a *bad* channel is rewarded with the possibility to send to the MAC layer a maximum number of back-to-back frames (hereinafter referred to as ‘‘TXburst’’), proportional to its forced silence period. The MAC layer will

then use the EDCF bursting capability² of the current 802.11e draft [18], to avoid contentions within the burst³.

The building blocks of our algorithm are the following:

- channel state estimation
- queue selection and service
- TXburst length setting.

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 is not needed at WSs. Below, we briefly describe these building blocks for the most general case, i.e., for the AP.

A. 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 [13]. 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

²Setting the TxOpLimit to the size of the burst received by the LLC layer.

³The EDCF bursting capability provides for contentions only for the first frame of the burst, while each subsequent frame in the same burst can be transmitted after the ACK of the previous frame without contention. If an ACK is not received, the burst transmission is aborted. This recalls the transmission technique of fragments in 802.11b.

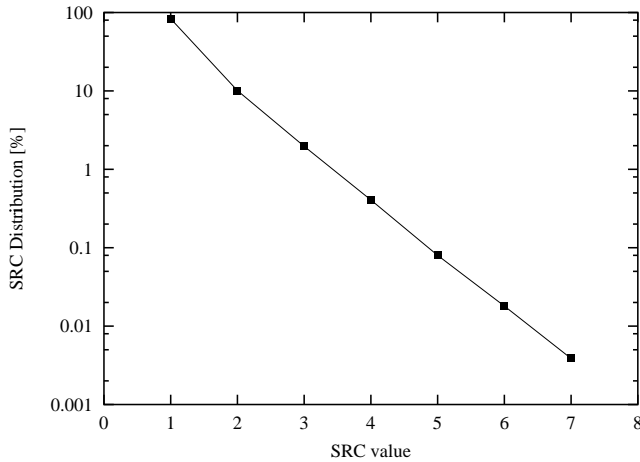


Fig. 3. Distribution of SRC instances

(namely, 40) and ideal channel conditions; as shown in Figure 3, the probability of SRC ever reaching values greater than 4 is almost negligible. 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 stored 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 single data frame (or RTS) to check the new channel state. The value of the PTIMER should be set small so as to quickly recover from short channel error periods (during long channel error periods, an excessive number of probing attempts is avoided by doubling PTIMER whenever the flag is reset to the BAD value after a probing period).

B. Queue Selection and Service

The process of queue selection and service at the LLC layer relies on indications from the MAC layer regarding the success or the failure of a frame transmission on the downlink. At the LLC layer, queues are served in a round-robin fashion. The queue selection algorithm is sketched in Figure 4, where two macro-blocks are outlined. The top one describes the actions taken when frames are sent to the MAC layer and the transmission outcome is notified to the LLC layer; the bottom one describes the queue selection.

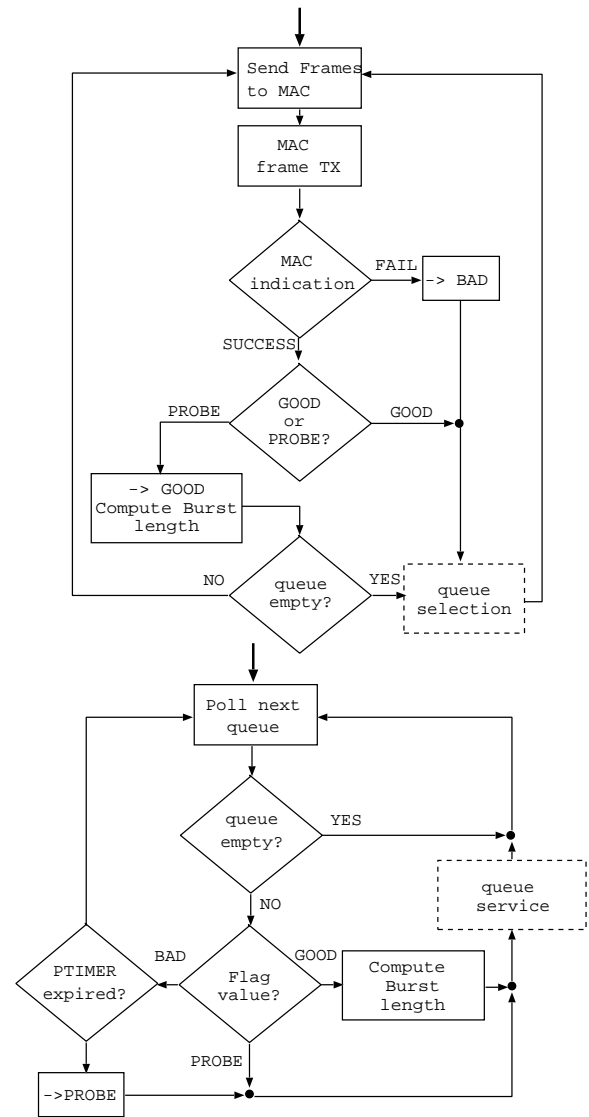


Fig. 4. Queue service (top) and queue selection (bottom) flow charts

The communication and data management between MAC layer and LLC layer occurs following the standard OSI paradigm of request/indication. The LLC layer sends a transmission request to the MAC layer along with the frames, belonging to the same TXburst, that are to be transmitted. These frames also remain in the LLC-layer buffer in case retransmission is required. The MAC layer then attempts the transmission following the EDCF 802.11e burst transmission technique. If all frames in the TXburst are successfully transmitted (i.e., the last MAC-layer ACK of the burst is received), the MAC layer returns a “frame success” indication to the LLC layer. If a transmission error occurs, the MAC layer discards the remaining frames in the TXburst; these frames are however stored at the LLC layer. The MAC layer then returns a “frame failure” indication to the LLC layer, together with the number of successfully transmitted

frames in the current TXburst, which can be discarded at the LLC layer.

When the MAC layer supplies a “frame success” indication to the LLC, the following cases may occur: (i) if the flag of the queue that is being served has a GOOD value, the next queue is polled; (ii) if the queue flag has a PROBE value, the flag is set to GOOD and the queue is enabled to start a TXburst whose length is determined according to the procedure outlined in Section IV-C. If the queue is empty, the next queue is polled.

When the MAC layer notifies a failure, the flag is set to BAD and the next queue is polled.

Polling a queue involves checking both whether it is empty and the value of the associated flag. In case of not empty queue, (i) if the flag 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. Likewise, (ii) if the flag is PROBE, the queue is served. Finally, (iii) if the flag is GOOD, the TXburst length is computed and the queue is served.

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.

C. TXburst Length Setting

Here we describe how to compute the TXburst length for the queues being served. Our goal is to provide a fair service to all queues at the AP and to all WSs. As mentioned before, in this work we choose to provide short-term fairness in terms of station throughput on the downlink. We will describe the procedure at the AP first, and we will then extend it to the WSs.

The AP has first to compute a fair target throughput (Thr_Fair), that is the throughput value representing the fair service share that each WS must receive at a given time. Since the offered traffic is typically time-varying, the Thr_Fair value may change rather quickly with time. We thus compute Thr_Fair by using a moving window mean with a configurable width parameter ($WinLen$), which represents the amount of history that is used to estimate Thr_Fair .

During this time interval, the AP computes two values:

- $Total_Thr$: the total throughput at the LLC layer, including both downlink and uplink traffic flows;
- $WS_Counter$: the number of active WSs, computed from the unique addresses observed at the AP

during $WinLen$ ⁴.

The Thr_Fair value can then be computed as:

$$Thr_Fair = \frac{Total_Thr}{WS_Counter}. \quad (3)$$

Furthermore, each LLC queue has associated with it a variable that keeps track of the throughput achieved by the corresponding WS on both the downlink and the uplink channel, during the last $WinLen$ period ($Queue_Thr(i)$, $1 \leq i \leq N$). Note that, to provide fairness among all uplink and downlink flows, we need to keep track of both channels.

Next, assume queue i is selected to be served, and the TXburst length needs to be assessed. The AP compares the values of $Queue_Thr(i)$ and Thr_Fair , and computes the following parameter $\Delta(i)$ as,

$$\Delta(i) = \frac{Thr_Fair - Queue_Thr(i)}{Thr_Fair}. \quad (4)$$

Normally, a default TXburst is allocated to every queue. However, if $\Delta(i)$ is positive, this means that the corresponding queue has achieved a throughput smaller than the fair target value. Therefore the queue will receive a larger TXburst in the current round. Otherwise, if $\Delta(i)$ is negative, it means that the corresponding queue has achieved a throughput larger than the fair target value. Hence, the queue will receive a smaller TXburst in the current round.

Several choices can be made regarding the TXburst lengths and the $\Delta(i)$ intervals. We chose a default TXburst of 3, and a TXburst allocation as a function of $\Delta(i)$ as in Table II. The choice of a default TXburst larger than 1 allows the fairness provision process to converge more quickly, thanks to a system of penalties (TXburst smaller than default) and rewards (TXburst larger than default) that re-balances the transmission opportunities within a few rounds. The intervals of $\Delta(i)$ were empirically chosen as the best in all simulated scenarios.

In order to allow WSs stations to apply the same “controlled” bursty transmission, they need the Thr_Fair value estimated by the AP, since WSs cannot compute the achieved global throughput. To solve this problem, the AP must include the value of Thr_Fair in control frames that are broadcast on the WLAN (for example, in the Beacon Frames). With this value, each wireless

⁴Computing the number of unique addresses at the LLC layer cannot rely upon MAC addresses, since they are removed before the frame reaches the LLC layer. It is therefore necessary for the LLC layer to inspect the IP source address carried in each frame in order to estimate the number of transmitting WSs on the uplink; while on the downlink, either the active LLC queues can be monitored, or the IP destination addresses can be recorded.

TABLE II
TXBURST LENGTH

$\Delta(i)$	TXburst length
< -0.03	1
$-0.03 < -0.01$	2
$-0.01 < 0.01$	3
$0.01 < 0.03$	4
> 0.03	5

TABLE III
BYTES TO SEND

Flow Length (bytes)		
119	529	4706
179	658	8015
251	948	13681
334	1650	26641
428	2861	284454

station can compute its own Δ as in (4), then choose the TXburst length according to Table II.

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.

The system is simulated for 2000 s before results are collected, in order to avoid transient effects.

A. Higher-layer Settings

The traffic offered to the network is generated by TCP connections exhibiting an On-Off behavior. The average duration of the Off period is a configurable parameter that we vary in our simulations: during these periods no segments are generated. Clearly, by tuning the value of the Off period, we act on the system load. When a connection is turned On, its duration depends on the amount of bytes to send: this value is uniformly chosen among the ones reported in Table III, that were derived by measurements of real Internet traffic [19]

A TCP connection reverts to the Off phase only when the sender agent receives the acknowledgment for the last TCP segment that it sent during the previous On period.

B. Transport-layer Settings

The TCP version used in our simulation is NewReno, which is supposed to improve TCP performance during Fast Recovery phases (i.e., it recovers from multiple losses within the same window without waiting for a timeout expiration). 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.

C. Data Link-layer Settings

As stated above, the bit rate of WSs and AP is 11 Mbps. Minimum and maximum Contention Windows are equal to 31 and 1023, respectively. 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 are never transmitted using the RTS/CTS handshaking. LLC queues at WSs are 50 data frames long, while they are 400 data frames long at the AP.

VI. NUMERICAL RESULTS

In this section we evaluate the performance of our scheme by comparison against the performance obtained by the standard scheme. In the plots we label our algorithm by ‘fair’ and the standard one by ‘standard’. In order to evaluate the importance of an adaptive choice for TXburst, in some plots we also consider a modification of our scheme in which TXburst is fixed and equal to 3; in this case, curves are labeled by ‘fixed TXburst’.

We first analyze the behavior of our scheme under variable traffic conditions by letting the number of WSs vary. We set the error channel model parameters according to Table I and the average duration of the TCP source Off periods to 1 s.

Figure 5 shows the average throughput achieved by a connection as the number of WSs, N , varies between 8 and 40. We assume for our scheme a value of $WinLen$ equal to 10. In the plots, curves labeled by ‘all’ refer to values of the throughput obtained by averaging over all connections; curves labeled by ‘UL’ and ‘DL’ refer, respectively, to uplink and downlink connections only. The solid lines refer to the standard, while the dashed lines correspond to the proposed scheme. Clearly, as N increases, the per-connection throughput decreases; the decrease is not proportional to N due to the increase with N of the collision probability. The proposed scheme achieves almost twice the throughput than the standard for any considered value of N . In terms of fairness between uplink and downlink, both schemes allow the

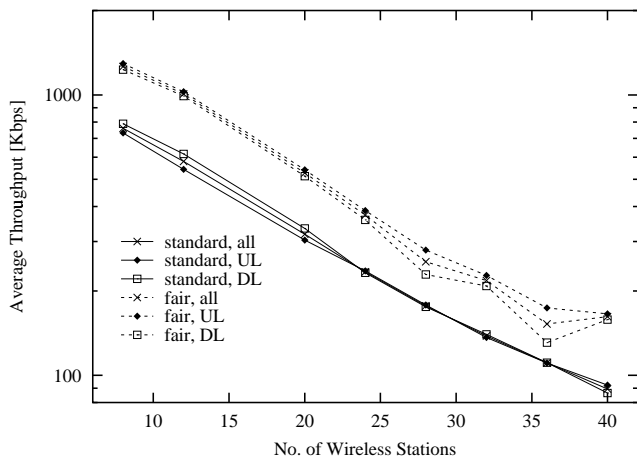


Fig. 5. Average per-connection throughput (average computed over all connections, only the uplink ones or only the downlink ones) versus the number of wireless stations

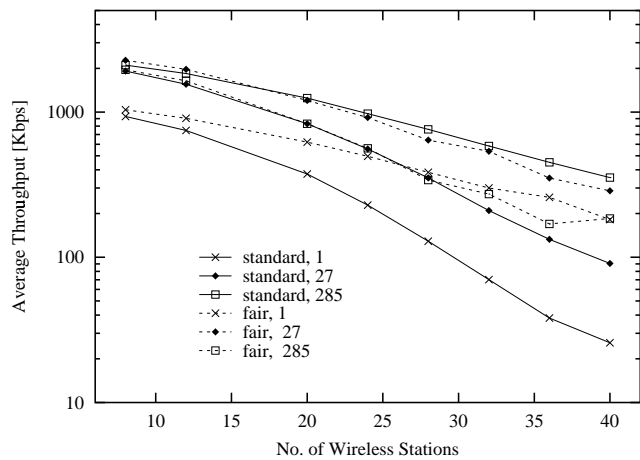


Fig. 7. Average throughput of connections with size 1, 27 or 285 segments versus number of wireless stations

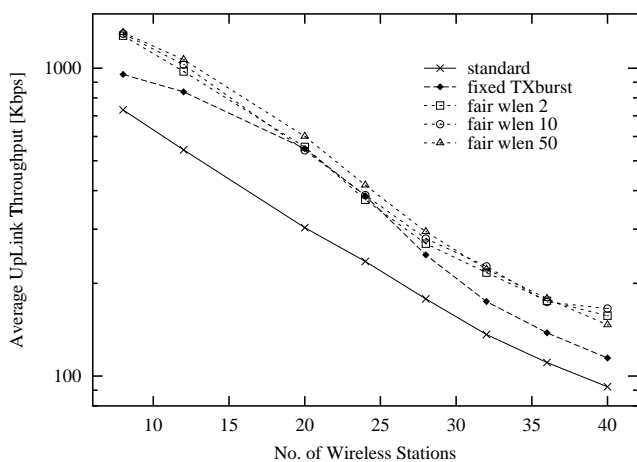


Fig. 6. Average throughput of uplink connections versus the number of wireless stations

uplink and downlink connections to equally share the bandwidth and achieve the same average throughput. Indeed, fairness between uplink and downlink connections is guaranteed by the closed-loop control of TCP whose throughput is regulated by the average round-trip delay which, by means of the forward stream of segments and the backward stream of TCP ACKs, includes both the uplink and the downlink delay. Despite the closed-loop control of TCP, some unfairness between uplink and downlink connections was observed in [17] under the standard scheme due to larger losses at the AP than at the WS queues. Since in our scenario the AP buffer is larger than the buffer at the WSs, losses at the AP are smaller or comparable to those at the WSs; therefore we do not observe this phenomenon.

Figure 6 reports the throughput achieved by the uplink connections with different values of the parameter

$WinLen$, which represents the time window over which the number of active connections is estimated. The parameter $WinLen$ does not have a remarkable impact on the results, thus implying that the activity of sources is high enough so that the number of WSs can be accurately estimated even with short windows. The importance of dynamically adapting the TXburst parameter can be assessed by observing the gap between the performance of the proposed scheme and the fixed TXburst one.

We next analyze the performance experienced by flows of different length. We consider very short connections which contain 1 segment only, medium-size connections containing 27 segments (these are medium-size files of almost 30 Kbytes) and very long connections with 285 segments. We highlight that, as can be seen from Table III, with 1000-byte-long TCP segments, more than 80% of the connections are limited to 1 segment. Figure 7 reports the average throughput versus the number of WSs. Solid lines refer to the standard, dashed lines to the proposed scheme with $WinLen = 10$. Under the standard scheme, long connections experience much higher throughput than short ones due to the behavior of TCP which starts by slowly loading the network looking for the optimal operating point: short connections are penalized since they will never reach the optimal operating point. The difference between long and short connections throughput is reduced under the proposed scheme; i.e., the variance of the throughput experienced by connections of different size decreases. This is a very interesting feature of our solution, which is due to the initial setting of the parameter $Queue_Thr$. When a connection starts, its estimated throughput $Queue_Thr$ is equal to 0 and the TXburst is maximum so that we slightly favor the initial phase of a connection. Doing

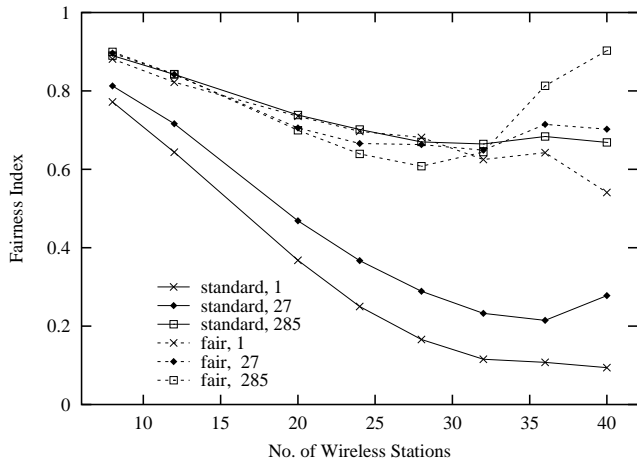


Fig. 8. Fairness index of connections with size 1, 27 or 285 segments versus number of wireless stations

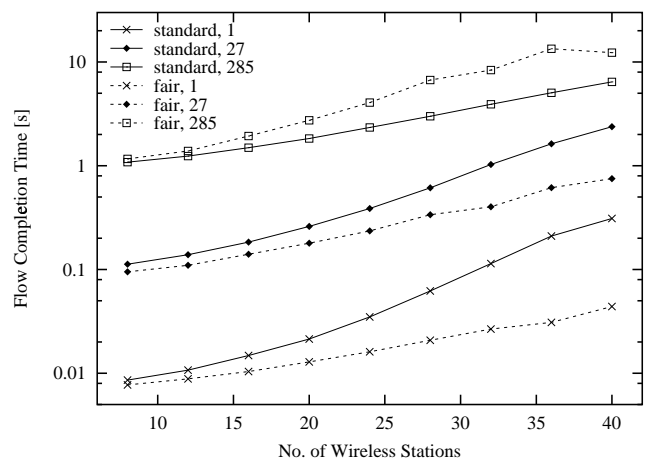


Fig. 10. Average connection completion time versus number of wireless stations

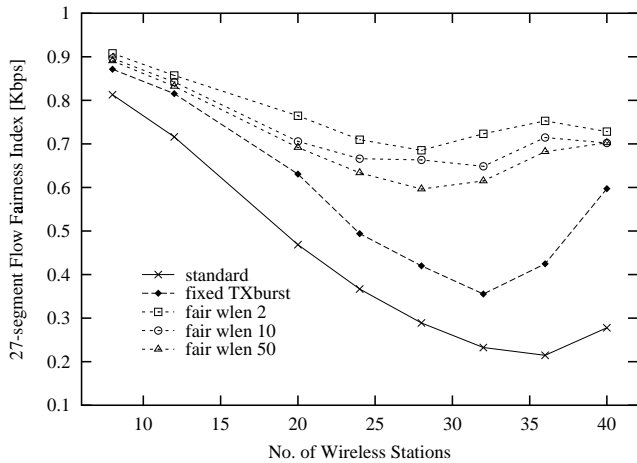


Fig. 9. Fairness index of connections with size 27 segments versus number of wireless stations

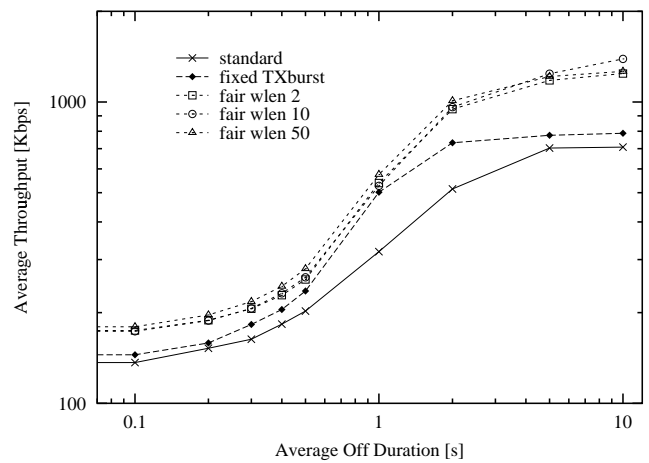


Fig. 11. Average throughput versus the average value of the source Off period

so, short connections are less penalized by the slow initial dynamics of TCP. Notice, however, that a different choice of the initial values of $Queue_Thr$ and $TXburst$ could better suit transport protocols other than TCP.

For the same scenario, Figure 8 shows the fairness among connections. As a metric of fairness we use Jain's fairness index [20]. The proposed solution outperforms the standard in fairness provisioning: fairness never drops below 0.7 in the worst case under our scheme, while it can be as low as 0.1 for the standard one. Notice, also, that the proposed scheme fairness is less sensitive to the number of Ws than the standard is.

The impact of $WinLen$ and of the dynamic adjustment of the $TXburst$ on the fairness index is shown in Figure 9 where only 27-segment-long connections are considered. While the choice of $WinLen$ does not significantly affect the fairness index, the dynamic setting of $TXburst$ is needed in order to achieve high

values of the fairness index.

Since the QoS perceived by end users is related to the time it takes to download a file, we show in Figure 10 the completion time of connections with 1, 27 and 285 segments. 90% of the connections, i.e., 1- and 27-segment connections, complete their transmission faster under our scheme than under the standard one with an obvious advantage for the user.

We now study the effect of the load on the performance by changing the traffic source behavior instead of the number of Ws. We set N to 20 and we let the TCP source Off periods vary. Figure 11 reports the per-connection throughput as the load decreases. The proposed scheme always achieves higher throughput than the standard. Notice also that, again, the parameter $WinLen$ does not have a remarkable impact on the throughput. This confirms that the connection activity is high enough to be easily estimated in a short time

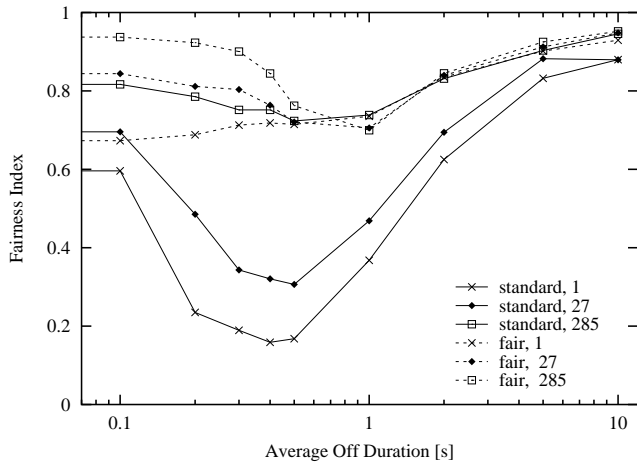


Fig. 12. Fairness index of connections with size 1, 27 or 285 segments versus the average value of the source Off period

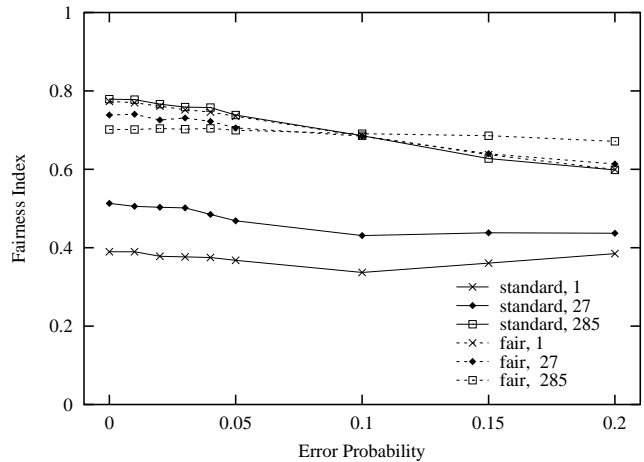


Fig. 14. Fairness index of connections with size 1, 27 or 285 segments versus the average slot error probability

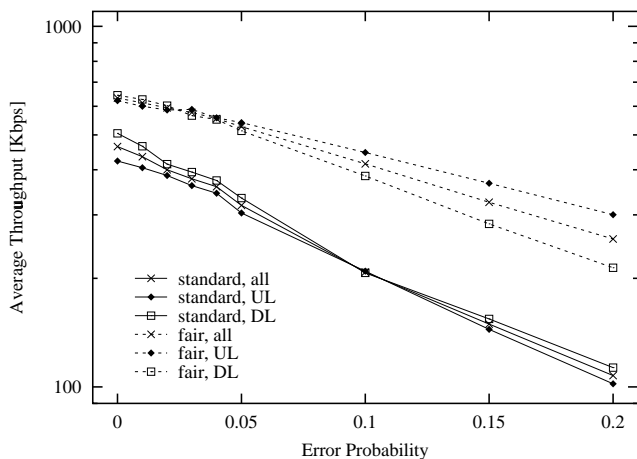


Fig. 13. Average per-connection throughput (average computed over all connections, only the uplink ones or only the downlink ones) versus the average slot error probability

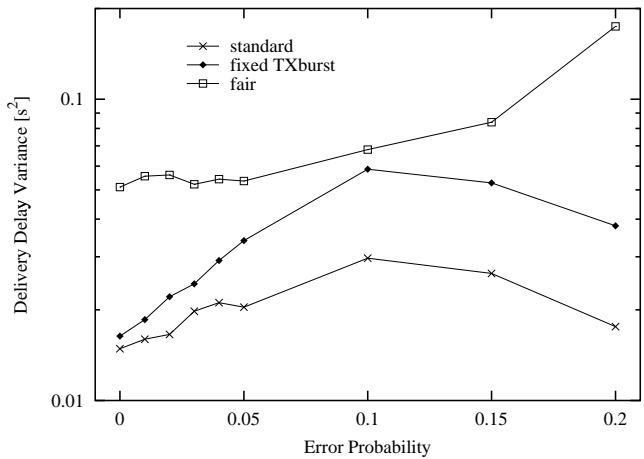


Fig. 15. Delivery delay variance versus the average slot error probability

window. On the contrary, estimating the number of active connections becomes difficult when Off periods are short; under these conditions the improvement achieved by the proposed scheme is slightly smaller than for long Off periods.

Figure 12 shows the fairness index versus the average source Off period when we consider connections 1-, 27- or 285-segment-long. Again, the improvement obtained by the proposed scheme is remarkable.

We assess the impact of channel errors on the system performance. We set the average duration of long error bursts, L_{LB} , and of short error bursts, L_{SB} , to the values reported in Table I and we change the average number of consecutive good slots so that a desired value for the average slot error probability is obtained. Figure 13 reports the average per-connection throughput versus increasing values of the average error probability.

Clearly, the throughput monotonically decreases with the error probability; however, the proposed scheme always obtains larger values of throughput than the standard. The gap between the performance of the two schemes increases with the average error probability, reaching a factor 2 for error probability equal to 0.2. The effectiveness of our solution for large values of the error probability, which are the most critical ones, makes the scheme particularly appealing.

The fairness index for connections of 1, 27 or 285 segments is shown in Figure 14 versus increasing values of the average error probability. Fairness is almost insensitive to errors over the channel, however, it is always larger for the proposed scheme than for the standard one. Moreover, connections of different size perceive almost the same fairness, while this metric is sensitive to the connection size under the standard scheme.

As a final remark, in Figure 15 we plot the variance

of the delivery delay (time elapsed since a frame is enqueued for transmission at the LLC layer, until it is correctly received at the destination) as a function of the average error probability. As can be expected from all scheduling mechanisms that privilege fair service over service in order of arrival, our scheme performs worst in term of delay variance, although the performance gap is small.

VII. CONCLUSIONS

The standard 802.11b dictates that, upon detecting poor channel conditions, a WS should attempt new transmissions after lowering its bit rate. While this solution may sometimes get around the temporary channel failure, it is pointed out in [2] that it may lead to performance degradation of the whole WLAN. The algorithm described in our paper does not resort to bit rate reduction upon channel failure, thus avoiding the anomaly, but it silences WSs while the channel is unavailable and uses a burst award mechanism to compensate for the missed transmission opportunities. This solution, beside offering channel-time compensation, is found to be highly beneficial for short-lived TCP flows, that normally suffer from losses on their early windows when competing with long-lived flows on a congested link. A large set of simulations confirmed this intuition and showed the superiority of our solution, with respect to the standard scheme, in terms of throughput, flow completion time and fairness.

REFERENCES

- [1] D. Black, S. Blake et al., An Architecture for Differentiated Services, *RFC 2475*, Dec. 1998.
- [2] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," *IEEE Infocom'03*, San Francisco, USA, Mar. 2003.
- [3] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *ACM Sigcomm'97*, Cannes, France, Sep. 1997.
- [4] S. Lu, T. Nandagopal, and V. Bharghavan, "A wireless fair service algorithm for packet cellular networks," *IEEE/ACM MobiCom*, 1998.
- [5] T.S. Ng, I. Stoica and H. Zhang, "Packet fair queueing algorithms for wireless networks with location-dependent errors," *IEEE Infocom'98*, Mar. 1998.
- [6] D. A. Eckhardt and P. Steenkiste, "Effort-limited Fair (ELF) Scheduling for Wireless Networks," *IEEE Infocom'00*, Tel Aviv, Israel, Mar. 2000.
- [7] G. Holland and N.H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," *IEEE/ACM MobiCom'99*, August 1999, pp. 219–230.
- [8] K. Tang, M. Correa, and M. Gerla, "Effects of ad hoc MAC layer medium access mechanisms under TCP," *ACM/Bal MONET*, vol. 6, no. 4, pp. 317–329, 2001.
- [9] C.L. Barrett, M. V. Marathe, D.C. Engelhart, and A. Sivasubramaniam, "Analyzing the short-term fairness of IEEE 802.11 in wireless multi-hop radio networks," *MASCOTS'02*, Fort Worth, TX, USA, Oct. 2002. Simulation of Computer and Telecommunications Systems.
- [10] J.L. Sobrinho and A. S. Krishnakumar, "Real-time traffic over the IEEE 802.11 medium access control layer," *IEEE Globecom'96*, Nov. 1996.
- [11] 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.
- [12] 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.
- [13] N. H. Vaidya, P. Bahl, and S. Gupta, "Distributed fair scheduling in a wireless LAN," *IEEE/ACM MobiCom'00*, Boston, MA, USA, Aug. 2000.
- [14] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11," *IEEE Infocom'01*, Anchorage, AK, USA, Apr. 2001.
- [15] A. Grillo and M. Nunes, "Performance evaluation of IEEE 802.11e," *IEEE PIMRC'02*, Lisboa, Portugal, Sep. 2002.
- [16] C. Koksal, H. Kassab, and H. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," *ACM SIGMETRICS'00*, Santa Clara, CA, USA, June 2000.
- [17] 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.
- [18] IEEE 802.11 WG, Draft supplement to Part 11: Wireless LAN medium access control and physical layer specifications: MAC enhancements for quality of service, IEEE 802.11e/Draft 3.0, May 2002.
- [19] *tstat* - <http://tstat.tlc.polito.it/>
- [20] R. Jain et al., "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems", DEC RR TR-301, Sep. 1984.