Understanding TCP fairness over Wireless LAN

Saar Pilosof[1], Ramachandran Ramjee[2], Danny Raz[1], Yuval Shavitt[3], and Prasun Sinha[2]

[1]The Technion	[2] Bell Laboratories	[3]Tel Aviv University
Dept. of Computer Science	101 Crawfords Corner Road	Dept. of Electrical Engineering
Haifa 32000, Israel	Holmdel, NJ 07733, USA	Ramat Aviv 69978, Israel
{psaar,danny}@cs.technion.ac.il	{ramjee,prasunsinha}@bell-labs.com	shavitt@eng.tau.ac.il

Abstract— As local area wireless networks based on the IEEE 802.11 standard see increasing public deployment, it is important to ensure that access to the network by different users remains fair. While fairness issues in 802.11 networks have been studied before, this paper is the first to focus on TCP fairness in 802.11 networks in the presence of both mobile senders and receivers. In this paper, we evaluate extensively through analysis, simulation, and experimentation the interaction between the 802.11 MAC protocol and TCP. We identify four different regions of TCP unfairness that depend on the buffer availability at the base station, with some regions exhibiting significant unfairness of over 10 in terms of throughput ratio between upstream and downstream TCP flows. We also propose a simple solution that can be implemented at the base station above the MAC layer that ensures that different TCP flows share the 802.11 bandwidth equitably irrespective of the buffer availability at the base station.

I. INTRODUCTION

Local area wireless networks based on the IEEE 802.11 standard are becoming increasingly prevalent with a current installed base of 15 million homes and offices. The focus is now turning to deploying these networks over hot spots such as airports, hotels, cafes, and other areas from which people can have untethered public access to the Internet.

As these networks see increasing public deployment, it is important for the service providers to be able to ensure that access to the network by different users and applications remains equitable. Since the majority of applications in today's Internet use TCP, we focus on the problem of TCP fairness in wireless LAN.

Fairness issues in wireless LANs have been studied extensively [1], [2], [3], [4]. However, most of these solutions involve changes to the Media Access Control (MAC) layer. This is impractical given the wide deployment of these networks. Also, while the focus of previous work has been on ensuring a particular QoS level for a given flow, we are interested in TCP fairness in the presence of both *uploads and downloads* i.e. in the presence of both mobile senders and receivers, which has not been considered by any prior work.

Consider a typical installation of a 802.11 based wireless network where the mobile hosts access the network through a base station or access point. Since the 802.11 protocol allows equal access to the media for all hosts, the base station and the mobile hosts all have equal access to the medium. If the mobile hosts are all senders or all receivers, then they each have equal share of the total available bandwidth. However, consider the case when there is one mobile sender and the rest are all mobile receivers. In this case, the base station and the mobile sender get equal access to the media. This mobile sender, therefore gets half of the channel bandwidth and the remaining half is equally shared by all the mobile receivers. Depending on the number of receivers, the sender can achieve several times the bandwidth of the receivers. Thus, the very equal access nature of the 802.11 media access protocol, when applied to the standard installation of access through a base station results in significant unfairness. This unfairness problem is compounded further in the case of TCP because of the greedy closed loop control nature of TCP and is the focus of our paper.

In this paper, we evaluate extensively through analysis, simulation, and experimentation the interaction between the 802.11 MAC protocol and TCP. We identify four different regions of TCP unfairness that depend on the buffer availability at the base station, with some regions exhibiting significant unfairness of over 10 in terms of throughput ratio between upstream and downstream TCP flows. We also propose a simple solution that can be implemented at the base station above the MAC layer. The solution ensures that different TCP flows share the 802.11 bandwidth equitably irrespective of the buffer availability at the base station.

The rest of the paper is organized as follows. In Section II, we present the overview of the problem of TCP fairness over 802.11 networks. In Section III, we present simulation results highlighting the four different regions of unfairness with respect to the base station buffer availability. In Section IV, we model the behavior of multiple mobile TCP hosts accessing the 802.11 network through a base station. In Section V, we discuss approaches for solving the fairness problem and present our solution. In Section VI, we review related work and finally in Section VII, we present our conclusions.

II. PROBLEM OVERVIEW

In order to illustrate the subtle interactions of TCP with an unfair 802.11 MAC protocol, consider the simple case of one mobile TCP sender and one mobile TCP receiver interacting with the wired network through a base station.

We conducted a series of performance tests on a commercial 802.11b network consisting of one base station and three mobile users. In all tests we had two or three mobile stations communicating to a server through the base station. Table I summarizes the throughput ratios we observed in the different settings with R_u representing the average TCP uplink

MTU	# of up flows	# of down flows	UDP flow	R_u/R_d	SD
1500	1	1	–	1.44	0.22
1500	2	2	–	1.58	0.23
1500	3	3	–	1.76	0.34
1500	4	4	–	1.80	0.27
1500	2	2	500/2ms	1.79	0.35
1500	2	2	1000/2ms	2.15	0.55
500	1	1	–	1.77	0.39
500	2	2	–	1.83	0.38
500	3	3	–	1.87	0.41
500	1	1	450/1ms	3.05	0.83
500	2	2	450/1ms	7.9	4.57

TABLE I

THE RATIO BETWEEN THE UP AND DOWN FLOW IN USING COMMERCIAL 802.11B

throughput and R_d representing the average TCP downlink throughput. The ratios presented in the table are the average of 5-10 runs and the standard deviation is presented in the last column.

One can see that even for the basic case of one mobile sender (upstream flow) and one mobile receiver (downstream flow), there is no equal sharing of the bandwidth with the sender receiving 1.44 times the receivers bandwidth. This is interesting since one might expect a commercial system to give higher priority to the base station such that there is a bias towards the downstream flow and *not* the upstream flow, given that the majority of applications today involve download rather than upload. Also note that, when we increase the number of flows, we see that the ratio also increases.

In order to test the sensitivity of this ratio to the base station buffer size, we would like to vary the buffer size on the wireless interface card at the base station. However, we did not have direct access to the interface card. Hence, we decided to use background UDP traffic, carefully spaced, to constrain the buffer available to the two TCP flows. The packet size (450-1000 bytes) and inter packet intervals (1-2 ms.) of the UDP traffic is shown in Table I. We find that the divergence between upstream and downstream throughput becomes much more severe in the presence of background UDP traffic. We also experimented with smaller Maximum Transmission Unit (MTU) values and found that the throughput ratio reaches as high as 8 in some cases. Thus, we find that as the buffer available to the TCP flows is decreased, the ratio of the uplink to the downlink TCP flow increases.

In order to gain a better understanding of the reasons for this behavior, we installed sniffers on the wireless interface and analyzed the captured packets. Note that since we used commercial Microsoft products, we did not have access to the kernel and thus we had to use scripts that compute an approximation of the TCP window size. It appears that in all cases the upstream TCP window size reaches its maximum size (determined by the receiver window size) but the downstream window size changes. Figure 1 presents the accumulated received sequence number and the approximated window size during the duration of the session for the downstream flow. Note that the throughput in the first 150 seconds is very low; the window increases at the beginning, when the upstream

flow is not yet using its full window (the upstream data is not plotted) but at some point the window drops (again this is not shown too well since we can only guess the window size and in this case there are many lost packets). One can clearly see, though, that the sequence number does not increase immediately which indicates a timeout period and a very small window. At time 150 seconds the upstream flow finished its upload and terminated. At this time we can see that the window increases and it remains between 9000 and 18000 bytes in the congestion avoidance region. This is probably due to the fact that the background UDP flow competes with it at the base station.
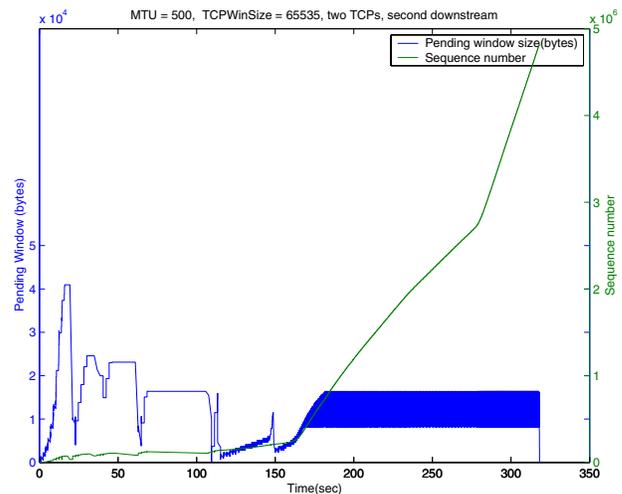
Fig. 1.   Downstream TCP flow with background UDP: approximated TCP window and throughput from testbed

While these experiments enabled us to verify our hypothesis of throughput unfairness between upstream and downstream TCP flows in 802.11 networks, a number of factors impact the throughput ratio in a test-bed. These factors include wireless link interference, base station buffer size, implementation details of the 802.11 MAC layer etc. Furthermore, it is difficult to obtain the values of some of these parameters (since it is typically not made public by the manufacturer) and it is impossible to isolate the impact of these parameters or study the impact of varying these parameters in a test-bed setting. In

order to carry a rigorous study of this problem, we therefore, use simulations instead of test-bed measurements. The results of the simulation study are described in the next section.

III. SIMULATION STUDY

In order to identify the relevant parameters and to analyze the up/down ratio we conducted a comprehensive simulation study using the NS2 simulator [5]. We start with the basic case of one mobile sender and one mobile receiver, and then consider the multiple flow scenario.

A. One upstream and one downstream flow
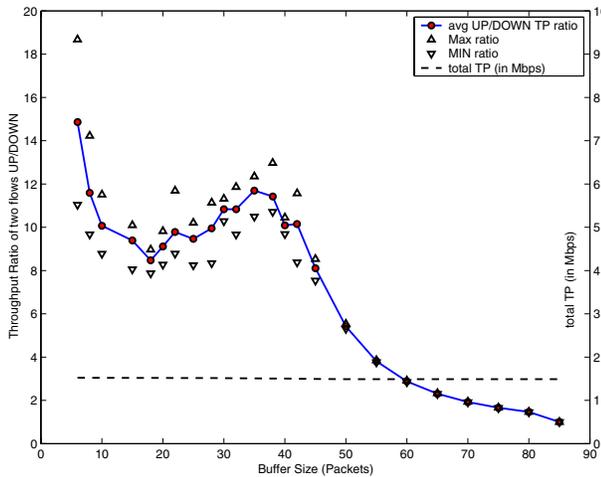
Fig. 2. One upstream and one downstream flow scenario: observed up/down ratio

In this case, we study the impact of the base station buffer size on the throughput ratio. We set TCP receiver window to 42 since in most commercial TCP implementations, the window size is set by default to 2^{16} which translates to about 42 packets, assuming an MTU of 1500 bytes. We vary the base station buffer size from 6 to 85. The results are shown in Figure 2. We also plot the total throughput in order to verify that it remains stable. For each buffer size, we conducted *5* simulation runs, each simulating *100* seconds of transmission. In addition to the average ratio, we also plot the maximum and minimum ratios, i.e., the maximum (minimum) ratio that was observed in any of the runs. The number of ACK packets per data packet (denoted by α) was set to 1 since in the most commonly used implementations of TCP this is the used default value. All data packets were of size 1024 bytes. In order to eliminate radio interference we placed the mobile stations at a fixed point close enough to the base station.

It can be observed from our results that the base station buffer size indeed plays a critical role in determining the ratio between the flows. There are basically *four distinguishable regions*. The first region corresponds to the case where the buffer size is over 84 packets and the throughput ratio is one. This reflects the case where the buffer is large enough to accommodate the maximum receiver window of both flows, thus resulting in loss-free transmission in both upstream and

downstream directions. The second region is when the base station buffer size is between 42 and 84 packets. In this region the ratio decreases sharply from 10 to 1. The third region corresponds to the case where the base station buffer size is between 6 and 42 packets and the ratio seems to vary between 9 and 12. The fourth region is when the buffer size is smaller than 6 packets. The results for this region are very noisy with the average serving as a poor representation of the dynamics. In Section IV, we analyze this behavior in more details using a simple model and explain why the ratio varies as shown in Figure 2.
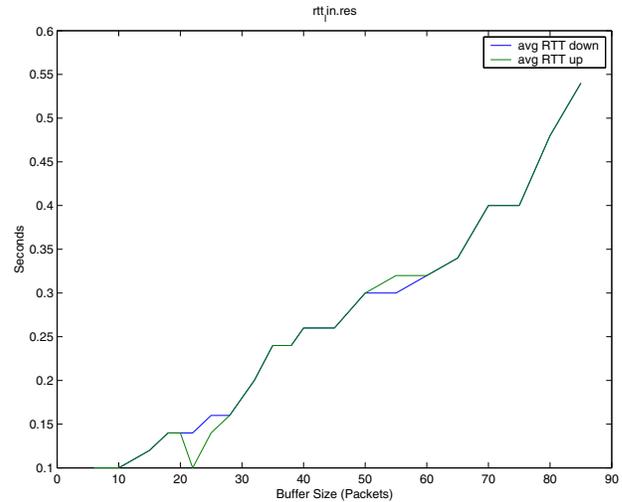
Fig. 3. One upstream and one downstream flow scenario: RTT values
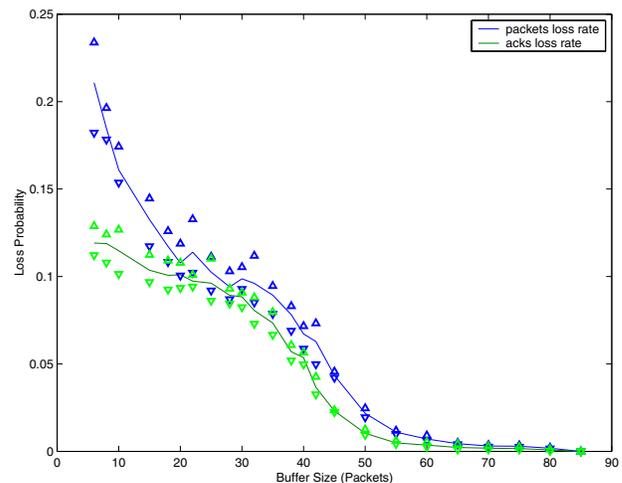
Fig. 4. One upstream and one downstream flow scenario: data and ACK loss rate

In order to gain a better understanding of the reasons for this behavior we also plot the average Round Trip Time (RTT) of both flows in Figure 3 and the loss rate for the data and ACK packets in Figure 4. One can see that the RTT increases monotonically with the base station buffer size without any significant rate changes. The loss behavior is a bit more

complex. To start with, the data packet loss rate is always higher than the ACK loss rate, and the dependency on the buffer size is not linear. We explain some of this behavior in Section IV.

In order to better understand the behavior of the wireless MAC layer, and the interaction with the TCP feedback mechanism, we plot in Figure 5, the accumulative throughput in packets sent by the MAC layer for each one of the stations. For the base station, we plot the ACK and data packets separately. Note that the information in this figure is accumulative, i.e., the wide dotted line indicates the total number of packets sent by the base station, and the difference between this line and the lower dashed line represent the amount of ACK packets sent by the base station. One can see that when the buffer size is smaller than 42 the relative share of each stream is almost fixed. This sharing results in the 1:10 ratio. When the buffer becomes larger the downstream traffic represented both by the downstream data packets and the corresponding ACK packets increases, which makes the ratio in figure 2 to decrease. This is reflected by the fact that when the base station buffer size is large (> 84), the height of the dashed line and the distance between the Down Ack and the Up packets lines is the same (about 600 packets).
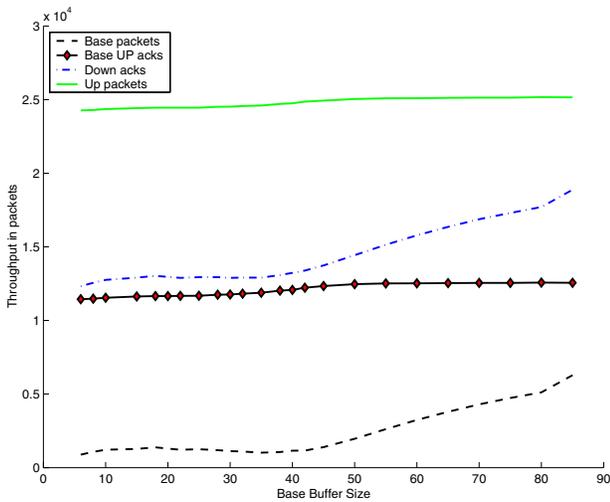
Fig. 5. Amount of packets sent through the MAC layer for different buffer sizes

B. Multiple flows

In this section, we consider several mobile users with multiple up/down flows. We consider two cases. First, we simulate the case of one upstream and multiple downstream flows and second, we simulate the case of equal number of multiple upstream and downstream flows. For these experiments, we fix the base station buffer size at 100 packets, modeling commercial 802.11 products. In these simulations each mobile host is sending or receiving one flow. Again we conducted 5 runs for each data point, lasting for 100 seconds of simulation time, and plot the average value.
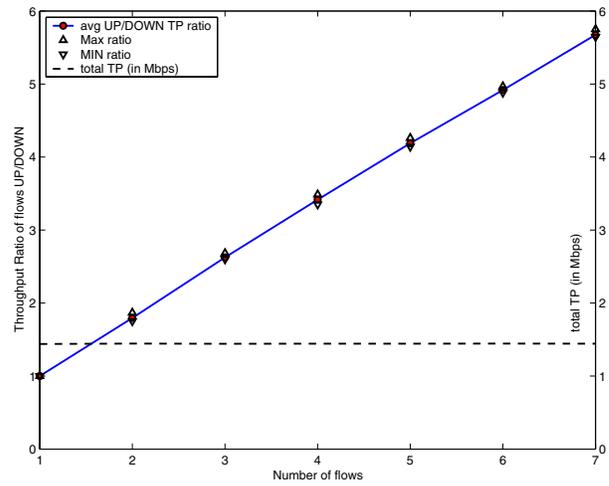
Fig. 6. Throughput ratio as a function of the number of downstream flows with one upstream flow

We plot the throughput ratio as a function of the number of downstream flows. In the first case(one upstream and multiple downstream flows) (Figure 6), we can see that the ratio is almost linear, i.e., all the downstream flows share the same resources while the total throughput remains stable.
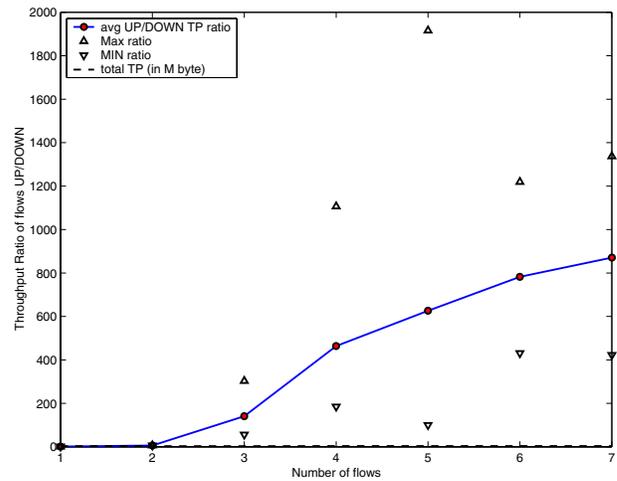
Fig. 7. The ratio as a function of the number of downstream flows with equal number of upstream and downstream flows

In the second case where we have equal number of multiple upstream and downstream flows, the situation becomes much more severe. In this case (see Figure 7) we can see average ratios of up to 800. This is due to the fact that the ACKs of the upstream flows clutter the base station buffer and as a result many of the downstream flows experience significant timeouts due to packet drops at the base station buffer, thus exacerbating the unfairness that is already present in the network. Again, as we show in the next section, even when the buffer is being cluttered and the ACK loss rate is high, upstream flows still arrive at the maximum window size, while downstream flows struggle with a window of 0-2 packets.

IV. Modeling TCP access

In order to understand the issues behind the observed unfair behavior of TCP over wireless LAN, and to try to develop tools that enable a more equitable usage of the bandwidth resources, we conducted an analytical study of the problem. We start with the simple case where there are only two users in the system, one sending data upstream and one retrieving data downstream.

A. One upstream and one downstream flow

As described in the previous section, the behavior in this case depends heavily upon the size of the buffer at the base station, denoted by B, and the TCP receiver window size, denoted by w. We assume that all losses in the system occur due to buffer overflows at the base station.

A basic observation, that is the first step towards understanding the behavior, is that when the window size is large enough (more will be said about what is "large enough" later in this section) a loss of an acknowledgment packet has no real influence on the sender window size. This is due to the *cumulative acknowledgment* nature of TCP whereby the next ACK packet will have the appropriate sequence number and make up for the loss of the previous ACK packet. Thus, the upstream TCP window size will increase until it reaches w, and will remain at that size throughout the duration of the connection (assuming no packet loss from other sources).

The downstream TCP window size, however, changes considerably, depending on B and w, since TCP reacts to loss of each data packet by halving its window (unlike the loss of ACK packets which have no effect on the upstream TCP source, as discussed above). Clearly if the base station buffer is larger than twice the TCP receiver window size (or more accurately $(\alpha+1)w$), all packets will have room in the buffer, and no packets will be dropped. In this case, the fair allocation of bandwidth to the three stations by the wireless MAC layer, guarantees a fair allocation of the bandwidth to the two TCP flows since the transmission time for ACKs is relatively very small. Indeed, if we look at Figure 2, we can see that when the base station buffer size is larger than 84 (twice the TCP receiver window size since as explained before α is one, both in our simulation and real traces) the ratio is one.

However, when for smaler values of B we can see that the upstream flow gets a larger share of the overall available capacity. A simplified explanation of the system's behavior in this case is the following. Consider the base station buffer in a steady state. It has at most αw ACK packets, and thus $B-\alpha w$ slots are available for the down link flow. Due to TCP's behavior in the congestion avoidance region, the average usage of this buffer will be $\frac{3}{4}$-th, since whenever the window size goes over the number of available slots, a packet is lost, this is detected by the sender when it detects three duplicate ACKs, and the window decreases to half of its value. Thus, the window size will vary between $\frac{B-\alpha w}{2}$ and $B - \alpha w$, and the average window size will be $\frac{3(B-\alpha w)}{4}$. In such a case the ratio between the downstream throughput and upstream throughput is given by

$$\bar{R} = \frac{4w}{3(B - \alpha w)}$$

This simple explanation predicts a value that is not too far away from the measured one when B is large (more than $1.5w$), but it does not provide a good explanation for smaller values of B (see Figure 8, the dot-dashed 'naive ratio' line). The main problem with it is that it assumes that the base station buffer is basically full with αw acknowledgment packets all the time. This is definitely not the case since for smaller B values most of the time there are significantly fewer acknowledgment packets in this buffer, and therefore there is much more room for data packets of the downstream flow.

We now focus on obtaining a more accurate model of the unfairness problem. One can model the base station buffer as a bounded size queuing system (M/M/1/K), of size B (assuming that a packet is cleared from the buffer only after it has been successfully transmitted). In this system the service rate is the rate (in packets per time unit) the base station is served by the wireless MAC layer, and the arrival rate is $R_d + \alpha R_u$, where R_d and R_u are the rates of the downlink and uplink TCP flows, respectively. The probability that such a queue in its stable state has exactly k packets in the buffer is given by [6, pg. 104]

$$p_k = \frac{1 - \rho}{1 - \rho^{k+1}} \rho^k, \tag{1}$$

where ρ is the ratio between the arrival rate and the service rate. Using $\alpha = 1$, we get

$$\rho = \frac{R_u + R_d}{R_u} = 1 + \bar{R}, \tag{2}$$

where $\bar{R} = \frac{R_d}{R_u}$. The drain rate is R_u because the rate that the base station gets is equal to the rate of the upstream since we can assume that both buffers are never empty (this cannot be said about the uplink acknowledgment buffer which may be empty at some times during the transmission). Plugging $(1 + \bar{R})^k \approx 1 + k\bar{R}$ which is valid for small enough $\bar{R}$, and Eq. 2 in the formula for p_B (Eq. 1), the drop rate p is approximated as

$$p = \frac{1 + B\bar{R}}{B + 1}. \tag{3}$$

However, both p and $\bar{R}$ are unknown at this stage. In order to obtain another relation between these two parameters we use the well known results of Padhye et al. [7] that approximate TCP throughput under various conditions. If we assume that no timeouts occur, we can use Eq. (20) from [7] and get $R_d = \frac{1}{RTT_d}\sqrt{3\alpha/2p}$, Where RTT_d is an average RTT of the downlink flow. We also know that $R_u = w/RTT_u$ since as explained before, the upstream flow is bounded by the receiver window size. Thus we have:

$$\bar{R} = \frac{RTT_u}{RTT_d}\sqrt{\frac{3\alpha}{2w^2 p}}. \tag{4}$$

Since most of the delay of both flows is due to waiting in the base station buffer, and it is equal, we will assume for now

that $\frac{RTT_u}{RTT_d} = 1$ (see Figure 3 which substantiates this). Using (3) and (4) we get:

$$\frac{1 + B\bar{R}}{B + 1} = \frac{3}{2w^2\bar{R}^2}$$

Solving this equation we get the following expression for $\bar{R}$ as a function of B and w:

$$\bar{R} = \frac{-1}{3B} + \frac{4 \cdot 2^{\frac{2}{3}} w^2}{(81\,B^2\,w^4 + 81\,B^3\,w^4 - 4\,w^6 + X)^{\frac{1}{3}} 12B}$$
$$+ \frac{2 \cdot 2^{\frac{1}{3}} \left(81\,B^2\,w^4 + 81\,B^3\,w^4 - 4\,w^6 + X\right)^{\frac{1}{3}}}{12Bw^2} \quad (5)$$

where $X = \sqrt{w^8 \left(-16\,w^4 + (81\,B^2 + 81\,B^3 - 4\,w^2)^2\right)}$.

One can now plot $\bar{R}$ as a function of B where w is set to be 42; this is the dashed 'Computed ratio' in Figure 8 in the region 6-42. In order to verify our calculation for the region we are interested in ($B \approx 42$, $\bar{R} \approx 1/10$) we can use $1 + B \approx B$ and $1 + B\bar{R} \approx B\bar{R}$, and get: $\bar{R} = (\frac{3}{2w^2})^{1/3} = \frac{1}{10.56}$. This means that in the interesting region $6 < B \leq 42$ the ratio is almost constant and about $1:10$, as reported by the simulation results.

One interesting question that arises is, can we use this latter analysis also in the region $B > 42$? From a first look it appears that there is no problem; the drop probability will decrease (by very little though) and this may cause R_d to increase. The reason this will not work is that the analysis in [7] assumes a uniform (with respect to the window size) loss probability. This is definitely not the case for our scenario. If $B > \alpha w$ and the only loss is due to buffer overflow, losses occur in the downlink flow *only* when the window size is large. Thus the effective window size for the downlink flow is composed from a fixed part of size $B - \alpha w$, and a part that reflects the interaction with the acknowledgments in the base station buffer. However, when losses occur, the window size drops to half of its previous value, therefore the effective window size is approximated by $\sqrt{\frac{3\alpha}{2p} + \frac{3(B - \alpha w)}{4}}$, and we get the following ratio.

$$\bar{R} = \frac{RTT_u}{w * RTT_d}\left(\sqrt{\frac{3\alpha}{2p}} + \frac{3(B - \alpha w)}{4}\right). \quad (6)$$

In this case we get a more complicated relation for $\bar{R}$. The predicted values of $\bar{R}$ are represented in the graph in Fig 8 indicated by the "computed ratio" curve in the range 42-85. One can observe that in this region the computed ratio indeed explains the observed values very accurately. Moreover, for $w = 42$ this curve matches the value described earlier for the 6-42 region since in this case $B - \alpha w = 0$.

Note that while our model produces an excellent fit with the simulation for the region with buffer size greater than 42, it only produces a reasonable fit in the 6-42 buffer size region and does not yet fully explain the variations present. One possible drawback of our analytical model is that the M/M/1/K assumes a "nice" arrival behavior. This is not the case when we consider TCP data packets. In some cases TCP

will generate 2 packets back-to-back. This situation occurs when the window size is increased by more than the MTU. In particular, in the congestion avoidance region, this back-to-back phenomena happens every 'window size' packets on the average.

For example, when the buffer size is 40, the average TCP window of the downstream flow is about 4 ($42/10$), and thus the data packet loss rate is in fact 1.25 times the ACKs loss rate. This fits well with the measured rate as reported in Figure 4 for this region. However, this observation does not provide a complete explanation for the micro dependency of the rate on B in this region. We are currently examining this issue in more detail.

B. Small Buffer

Now, consider the case where the available buffer size for each flow is very small. We want to evaluate the upstream flow in this case. As mentioned earlier, TCP reaction to a loss of a number of acknowledgment packets can be either getting into a timeout, or increasing the window until it reaches the receiver window size. This is due to the fact that acknowledgment loss cannot result in three duplicate ACK as the acknowledgment number in each new ACK packet is different (assuming no data packet loss). Therefore, when the base station buffer size becomes very small (1-2 per flow) the connection throughput becomes very chaotic.

To better understand the state where the system spends most of its time in this situation, we can use the discrete time Markov chain of Figure 9. Each state in the Markov chain represents the window size of the uplink TCP sender. We only considered exponentially increasing steps, and thus state i represents a state where TCP window size is 2^i (state 0 represents a window size of 1 etc.). Once in state i, we can either go to state 0 (i.e. to window size 1) if a timeout occurs (it happens only if all ACK packets are lost, and the probability for that is p^{2^i}), otherwise we double the window size thus moving to state $i + 1$ with probability $1 - p^{2^i}$. Intuitively, one can expect the system to be working with full window size or in the reset state (state 0) as explained below.

Once a connection reaches a full window size it needs to loose w ACK packets to reset the window size. On the other hand, if the window size is very small, say 2, we only need to loose two packets to reset the counter. An exact analysis of the Markov chain of Figure 9 shows that the system spends almost all its time working at full window size, many orders of magnitude more than in all the other states combined. The exact difference depends on p. In line with our initial intuition, the analysis shows that if the system is not working with a full window, it is most likely working with window size equal to 1, i.e., in the reset state. Note, that the above model does not capture the full behavior of the TCP connection since there are issues involving doubling the initial timeout window and eventually flows may just give up and close connection. This is the reason for the very noisy results we get when the window size per flow operates at small values. We note that we tested this scenario with a buffer of 2 through NS2 simulations and
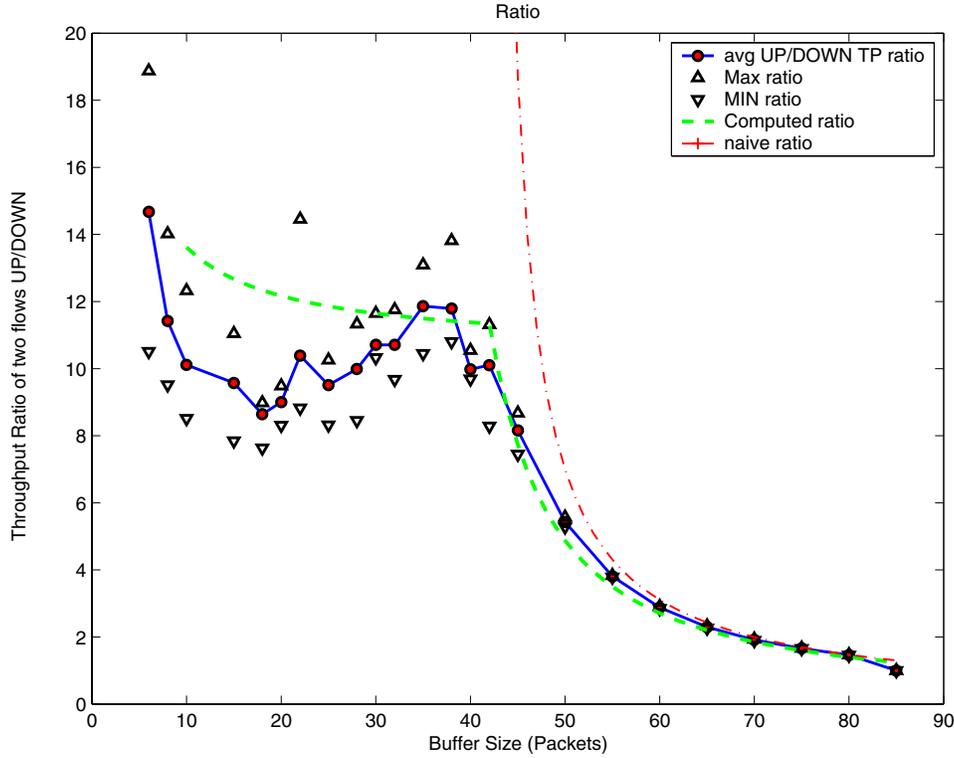
Fig. 8.   Analysis versus simulation results
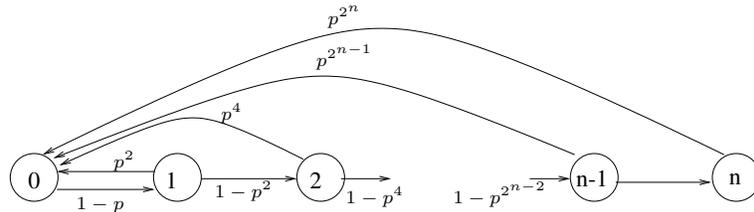
Fig. 9.   Markov chain

the upstream flow always ends up with the maximum window size.

C. Multiple flows

For multiple downstream flows, we can say that the same amount of "free" buffer space is divided among all n downstream flows, and therefore each one gets $1/n$ of the bandwidth and the ratio increases by a factor of n. However, this is not completely true since the utilization of the buffer space is better when a number of flows are involved. This explains the almost linear behavior of Figure 6. In order to get a more precise explanation of the ratio when the buffer is small, consider again Eq. 1. For this case ρ is give by

$$\rho = \frac{R_u + nR_d}{R_u} = 1 + n\bar{R}, \tag{7}$$

where we assume that all downstream flows get the same average rate $E[R_d]$, and $\bar{R} = \frac{E[R_d]}{R_u}$. Again, if $n\bar{R}$ is small enough we can use $(1 + n\bar{R})^k \approx 1 + kn\bar{R}$ and the drop rate

p is approximated by the following formula.

$$p = \frac{1 + nB\bar{R}}{B + 1} \tag{8}$$

Eq. 4 is still valid, and thus, in this case, we get the following equation:

$$\frac{1 + nB\bar{R}}{B + 1} = \frac{3}{2w^2\bar{R}^2}$$

This region, however, is not the one shown in Figure 6, since the region there represents $B = 100, w = 42$. For this case we should use Eq. 6; in this case the ratio between the upstream flow and each of the n down stream flows is expressed by the following formula.

$$\bar{R}_n = \sqrt{\frac{3\alpha}{2npw^2} + \frac{3(B - \alpha w)}{4nw}} \tag{9}$$

In this case, for small n, the available buffer is actually more than the receiver window. Figure 10 plots the ratio $\bar{R}_n/\bar{R}_1$ for $B - \alpha w = 100 - 42 = 58$ and $p = 1/100$. One can see that

the analysis nicely matches the observed behavior from the simulation.
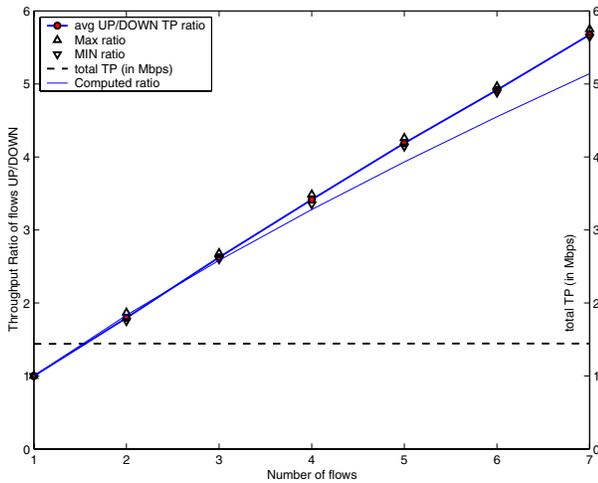
Fig. 10.   One upstream and n downstream flows: observed and computed ratio

V. OUR SOLUTION

In this section, we are interested in a solution that results in upstream and downstream TCP flows having an equal share of the 802.11 wireless bandwidth (throughput ratio of 1). The solution needs to operate above the MAC layer since changes to the MAC layer could involve expensive hardware upgrades given the wide deployment of 802.11 networks.

We first consider a simple intuitive solution of having separate queues for TCP and ACK packets at the base station with T packets for TCP data and A packets for TCP ACKs. Based on the discussion in the previous section, since dropping of several ACKs can not result in the TCP sender to back off due to the cumulative ACK feature of TCP, sizing of the ACK buffer, A, to ensure fair access to upstream and downstream flows becomes impossible to predict. The most we can do is create a timeout in this connection periodically in order to reduce the uplink utilization. This solution is clearly not effective.

Another feasible solution is to fake duplicated ACK packets thus forcing TCP to reduce the up stream window size. Alternatively, we can discard data packets for this flow (in the upstream direction). This solution will indeed reduce the window size but it is complicated. More importantly, this scheme wastes bandwidth as it either deletes data packets that already have been transmitted, or creates more duplicated ACK packets that use the limited resources (bandwidth and buffer space).

Our solution is to use the advertised receiver window field in the acknowledgment packets towards the TCP sender. This field represents the available space at the receiver and lowering the receiver window can help throttle the TCP sender. Thus, by manipulating the receiver window at the base station, we can ensure that the TCP sender window is limited to what ever value we decide. A similar approach was used in [8] for improving TCP performance over interconnected ATM and IP networks.

If there are n flows in the system and the base station has a buffer of size B, we set the receiver window of all the TCP flows to be the minimum of the advertised receiver window and $\lfloor B/n \rfloor$. This is performed by modifying the receiver window field of the ACK packets flowing through the base station. Note that this approach makes intuitive sense since if the base station (bottleneck node) is unable to buffer packets for the TCP source, it is better to throttle the source than drop packets. Also note that this approach accommodates different buffer sizes and number of flows and tries to deliver equitable bandwidth to all flows.

In order to implement such a solution one needs to keep a counter that approximates the number of current TCP flow in the system. Knowing the exact number of active flows may be problematic since some open connections may be actually idle. A more problematic point is that in many cases it is hard to determine whether the TCP connection is up stream or down stream, since connection may carry data in both directions. However this is not an issue for us since, regardless of the direction, we count each TCP flow (identified by the IP addresses and the port numbers) as a valid flow.
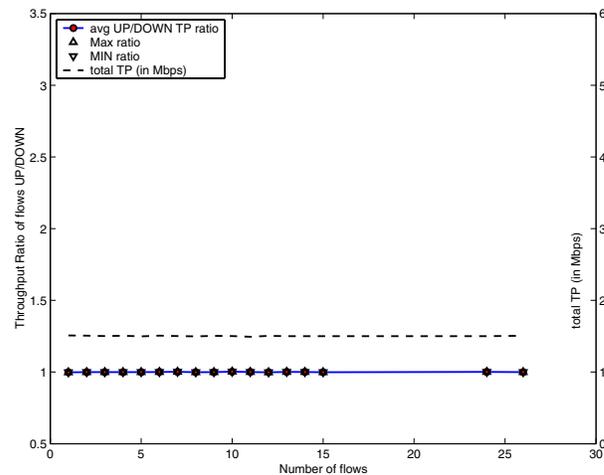
Fig. 11.   Throughput ratio when using receiver window manipulation

In order to verify if this approach delivers fair share to TCP flows, we performed simulations with varying buffer sizes and multiple number of upstream and downstream flows and computed the throughput ratio. In our simulation we use a base station buffer size of 100 packets; we simulated 5 runs for each n, each run lasting 100 seconds. Before each run we set the receiver window to be $\lfloor 100/n \rfloor$ Figure 11 plots the throughput ratio of upstream and downstream flows and shows that a 1:1 ratio is maintained, resulting in fair allocation of bandwidth. Furthermore, also note that the total throughput is maintained as the number of flows increases substantiating the fact that the overhead of this approach is minimal.

The impact of this very simple solution is made clear when we recall that in the same scenarios earlier, without the

modified receiver window solution, we saw ratios of 1:800 (see Figure 7). This is explained by the fact that the large number of ACK packets do not flood the buffer any more and each flow gets its fair share of the buffer space. Note that we increased the number of flows up to 26 in order to verify that even when the receiver window is set to the value of one packet, we still experience fair allocation of the bandwidth without any noticeable reduction in the total throughput.

In order to check if this solution works well in the real environment, we ran again some of the tests reported in Table I, but this time we set TCP receiver window at all receivers to a smaller value. In particular, when we used UDP background traffic with an MTU of 500 bytes, two upstream and two downstream flows, and we set the receiver window size to be 2000 bytes (instead of the default 65000) we observe a ratio of 1.007 with standard deviation of 0.0005 (compared to a ratio of 7.9 with default receiver window size). This is a strong indication that the proposed solution indeed works in commercial settings.

VI. RELATED WORK

Fairness over 802.11 based Wireless LANs (WLANs) has been studied by several researchers. Lu *et al.* [2] were the first to identify the problem (under a UDP traffic model) of fairness among users in a wireless LAN. Their solution, to the problem was a centralized scheduling algorithm to be performed at the BS. In addition, their solution required a special MAC algorithm where slots for transmission are specifically allocated to the other stations based on scheduling algorithm. Nandagopal *et al.* [3] suggest a fairness model that also identify the difference between node fairness and flow fairness. The model is used to compare the fairness achieved by different backoff mechanisms.

Another line of research [9] suggests to employ bandwidth reservation over MA channels in order to support quality of service (QoS). This approach can be suited for flows that have specific QoS requirement.

Several researchers have also proposed new MAC layers to provide fair channel access. Sobrinho and Krishnakumar [10] suggested a scheme called *blackburst* where channel jamming is used to find the real-time sender with the longest waiting time (and thus the higher priority). Deng and Chang [1] suggested to change the backoff period according to a station priority. The lower the priority the higher is the maximum backoff period a station can draw. Barry *et al.* [11] followed this line and suggested to use two distinct backoff periods for two priority classes. Vaidya *et al.* [4] suggested a clever distributed algorithm that calculates the backoff period for the stations such that the resulted access to the channel will closely match the Self-Clocked Fair Queuing (SCFQ [12]) scheduling. Recently, Aad and Castelluccia [13] suggested three differentiation mechanisms based on scaling of the congestion window, modifying the IFSs, and changing the maximum frame length.

However all these studies were either focused on UDP traffic or the fairness of MAC layer in isolation. Moreover, none of these papers present the effect of available buffer at the base station and the interaction of 802.11 MAC protocol on the user level unfairness observed at the TCP layer. Research on interaction of TCP over 802.11 based ad hoc networks [14], [4], [15] has taken factors such as mobility and multiple hops into account. However, the unfairness problems in 802.11 based WLAN installations arising due to buffer size availability has neither been studied nor observed before.

VII. DISCUSSION AND CONCLUSION

In this paper we presented fairness issues in 802.11 networks for TCP flows, and extensively evaluated the interaction between the 802.11 MAC protocol and TCP through analysis, simulations (on *ns2*) and experimentation. We found that the buffer size at the base station plays a key role in the observed unfairness. Based on simulations, we observed that the unfairness in TCP throughput ratio between upstream and downstream flows could be as high as 800. In our experiments, we were able to easily create simple scenarios exhibiting throughput ratio of about 8 times among TCP flows. Using a bounded size queuing system (M/M/1/K) we explained TCP's behavior and interaction with the MAC layer. The analysis identified four regions of TCP unfairness that depend on the buffer availability at the base station. Our proposed solution for alleviating the unfairness problem that uses advertised window manipulation at the base station, was tested on the simulator and in our testbed. It was shown to provide fair TCP throughput for any available buffer size or number of flows at the base station. Through our analysis we have been able to explain most of our TCP unfairness observations. However, there are still several open avenues that we are currently pursuing, some of which are as follows:

- Channel losses: In our simulations and analysis, we have assumed that the channel is error free. However, a lossy channel may result in packet drops due to channel error. In addition, link layer reliability mechanisms may introduce additional delays that may affect the fairness. Our analysis model needs to be extended to take these factors into account.
- TCP flows with different RTT: In our experiments as well as simulations, all the flows terminated at the same point in the wired network. This resulted in equal RTT for all flows and also helped in simplifying our analysis. However, the unfairness behavior can be different from the prediction based on our model if the flows have different RTTs. The analysis can be extended to take different RTTs into account.
- Providing higher share of the media to the BS: Our analysis is based on node level fairness provided by the 802.11 MAC protocol. The TCP unfairness behavior will be different if a MAC layer that provides user level fairness [1], [11], [4], [13] rather than node level fairness is used. Analysis of TCP behavior with such solutions and augmenting our proposed solution is part of ongoing work.

- Interaction with IPSec: The solution proposed in this paper cannot be used if the flow uses end-to-end IPSec since the transport headers will not be visible to the intermediary. This limitation is also true for all performance enhancing proxies, which are especially critical for wireless networks where bandwidth is a scarce resource. One way to tackle this issue is to use a split security model where the end hosts using IPSec trusts parts of the payload (such as transport headers) with the network intermediary. We are currently investigating this issue.

REFERENCES

[1] 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.

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

[3] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks," in *ACM Mobicom 2000*, Boston, MA, USA, Aug. 2000.

[4] N. H. Vaidya, P. Bahl, and S. Gupta, "Distributed fair scheduling in a wireless LAN," in *MobiCom 2000*, Boston, MA, USA, Aug. 2000.

[5] K. Fall and K. Vardhan, "*ns* notes and documentation," The source code and installation information available at http://www-mash.cs.berkeley.edu/ns/, 1999.

[6] L. Kleinrock, *Queueing Systems, Volume 1: THEORY*. John Wiley and Sons, 1973.

[7] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: a Simple Model and its Empirical Validation," in *ACM Sigcomm 1998*, 1998.

[8] L. Kalampoukas, A. Varma, and K. K. Ramakrishnan, "Explicit window adaptation: a method to enhance tcp performance," *IEEE/ACM Transactions on Networking*, vol. 10, no. 3, pp. 338–350, 2002.

[9] R. Yavatkar, D. Hoffman, Y. Bernet, F. Baker, and M. Speer, "SBM (subnet bandwidth manager): A protocol for RSVP-based admission control over IEEE 802-style networks," May 2000, internet RFC 2814.

[10] J. L. Sobrinho and A. S. Krishnakumar, "Real-time traffic over the IEEE 802.11 medium access control layer," *Bell-Labs Technical Journal*, vol. 1, no. 2, pp. 172–187, Autumn 1996, appeared also in Globecom'96, Nov. 1996.

[11] M. Barry, A. T. Campbell, and A. Veres, "Distributed control algorithms for service differentiation in wireless packet networks," in *INFOCOM'01*, Anchorage, AK, USA, Apr. 2001.

[12] S. J. Golestani, "A self-clocked fair queuing scheme for broadband applications," in *INFOCOM'94*, Toronto, Canada, June 1994, pp. 636–646.

[13] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11," in *INFOCOM'01*, Anchorage, AK, USA, Apr. 2001.

[14] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proceedings of IEEE/ACM MOBICOM '99*, August 1999, pp. 219–230. [Online]. Available: citeseer.nj.nec.com/holland99analysis.html

[15] K. Tang, M. Correa, and M. Gerla, "Effects of ad hoc mac layer medium access mechanisms under tcp," *MONET*, vol. 6, no. 4, pp. 317–329, 2001.