Throughput Unfairness in TCP over WiFi

Vasileios P. Kemerlis, Eleftherios C. Stefanis, George Xylomenos George C. Polyzos
Mobile Multimedia Laboratory
Department of Informatics
Athens University of Economics and Business
Patision 76, Athens 104 34, Greece
Email: vpk@cs.aueb.gr, leste@aueb.gr, xgeorge@aueb.gr and polyzos@aueb.gr

***Abstract*— This paper presents a measurement study of TCP performance at an operational WiFi deployment. After presenting the network topology and the tools used to generate and analyze traffic, we examine the throughput performance of competing TCP connections. We investigate how throughput is divided among the participating wireless hosts with respect to signal strength, traffic direction and use of the RTS/CTS mechanism. Our study shows that while competing clients with comparable signal strength are treated fairly, achieving similar throughput values, clients with lower signal strength are treated unfairly, relinquishing a larger share of the available bandwidth to clients with higher signal strength.**

I. Introduction

The IEEE 802.11b standard for *Wireless LANs* (WLANs), also known as *WiFi*, is becoming increasingly popular worldwide for providing wireless Internet access in University campuses and many other public areas. WiFi client devices (WLAN network interface cards) are now becoming standard equipment for mobile devices such as laptops, PDAs and advanced cell phones. Moreover, WiFi infrastructure devices (Access Points or APs) are increasingly used even in households, providing wireless coverage for home networks. This popularity, along with its easy and cheap deployment, indicates that WiFi technology will be an integral part of any Wireless on Demand System with aspirations to appeal to the masses. Such systems will have many users with diverse needs and/or access rights.

The WLAN environment is characterized by the existence of multiple wireless clients competing for a share of the bandwidth. *Quality of Service* (QoS) tools (such as [1]) can be used to allocate a fixed share of bandwidth to each client. However, in order to apply efficient QoS policies for these clients, it is useful to be able to estimate the throughput that a client is likely to achieve for a given amount of bandwidth.

The objective of this paper is to investigate how TCP throughout is affected by various network parameters by measuring the throughput values achieved by the clients. The parameters examined in this study are: i) signal strength and signal to noise ratio, determined by wireless card gain and topological factors, such as distance from the Internet *Access Point* (AP) and interfering obstacles, ii) RTS threshold value, which effectively enables or disables the RTS/CTS mechanism and iii) traffic direction, classified as either uplink (from client to AP) or downlink (from AP to client).

Our study is not concerned with improvements of TCP performance over WLANs. Measurements so oriented can be found in [2] and [3]. Our work only aims to discover the rules governing throughput behaviour in WiFi, so that they may be used for purposes such as client classification in WiFi specific QoS policies. We focus on the behaviour of TCP as it carries the majority of the traffic encountered on our campus WLAN, most other WiFi deployments and, of course, the Internet (e.g., see [4] and [5]; while UDP based measurements reveal more about the underlying network, it is TCP performance that most users experience. Our major conclusions are:

1) Throughput is unfairly distributed between competing TCP connections experiencing unequal signal strength, at least in a max-min sense of fairness. “Strong” clients dominate the wireless medium, forcing “weak” ones to drop their transmission rate below their fair share.
2) The RTS/CTS mechanism generally has a detrimental effect on maximum achieved throughput, but it may help to reduce the unfairness in some circumstances.

In the remainder of this paper we first describe the measurement testbed, including the hardware and software used and the procedures employed, and then proceed to examine single connection measurements, which serve to establish the base relationship between the throughput of a TCP connection and the parameters under study. We use these measurements as a baseline for our multiple connection measurements, where we investigate the achieved throughput per client when two clients are simultaneously transmitting or receiving TCP traffic.

II. Testbed Setup

A. Hardware Setup

The measurements took place in the operational IEEE 802.11b WLAN deployed in the *Computer Science Lab* (CSLAB) on the 2nd floor of the main building of the *Athens University of Economics and Business* (AUEB). The WLAN infrastructure consists of a Cisco 1200 IEEE 802.11b AP directly connected to a Linux based router (*Zeus*), allowing wireless nodes to access the Internet. We employed two (mobile) clients for our experiments, *Ares* and *Apollo*; one or both of them are active in the WLAN during tests. In order to limit the number of parameters that could influence performance, both clients used the same type of wireless network interface card. We used the Zoom Air 4100 PCMCIA 802.11b wireless cards which are fully compatible over the air

TABLE I
DESCRIPTION OF HARDWARE EMPLOYED

Name	Processor	Network Interface
Zeus	Pentium III 450Mhz	802.11b AP via IEEE 802.3 100BaseTX
Ares	Mobile Pentium 1700Mhz	PCMCIA 802.11b – Zoom AIR 4100
Apollo	Mobile Pentium 1600Mhz	PCMCIA 802.11b – Zoom AIR 4100

TABLE II
DESCRIPTION OF NETWORK SETTINGS

Parameter	Value
ESSID	AUEB
Channel	1 (2412 MHz)
RTS Threshold	200 bytes (ON) / 2346 bytes (OFF)
Fragmentation Threshold	2346 (OFF)
Radio Preamble	Short

with the AP, support nominal bandwidths of 1, 2, 5.5 and 11 Mbps and are equipped with a 2.2dbi dipole external antenna and the Prism I chipset. We confirmed that the performance of both our test hosts was symmetric by exchanging Ares' and Apollo's roles during our tests; results were almost identical. Table I shows the names and characteristics of each host. Ares and Apollo are laptops, while Zeus is a desktop machine.

Since Zeus acts as router, it has two identical IEEE 802.3 100BaseTX interfaces, one for Internet connectivity and one for connectivity with the AP. We used the `mii-tool` [6] to downgrade the interface between the AP and the router to IEEE 802.3 10BaseT, as preliminary tests showed that the fast Ethernet interface negatively affected the performance of the wireless LAN. This was due to congestion arising at the AP queue when forwarding packets from the (100Mbps) wired LAN to the (11Mbps) wireless LAN. Therefore, in order to avoid packet drops at the AP queue, we forced the wired LAN to a speed comparable to that of the WLAN.

All wireless client interfaces were set to operate at 11Mbps. We disabled the 802.11b rate adaptation mechanism in order to avoid unpredictable effects and test the native performance of TCP at a specific data rate. It should be noted that the rate adaptation mechanism seems to have unfairness issues of its own [7], [8]. Table II shows the network settings used during our measurements. For each experiment performed, all hosts used the same settings.

B. Software Setup

All hosts ran the Debian GNU/Linux operating system [9], kernel version 2.4.27, using the `orinoco_cs` WLAN drivers as kernel modules. Preliminary tests showed that the newer 2.6.x kernels face performance problems due to major changes in the networking subsystem; therefore, we reverted to a more stable kernel version. Linux was selected in order to enable the use of the many freely available measurement tools and to provide us with full control of all network parameters. All hosts were in multiuser mode during tests, but no user tasks were executing on Ares and Apollo. Zeus was running the appropriate Network Address Translation / Firewall rules.

TABLE III
DESCRIPTION OF MEASUREMENT LOCATIONS

Location	Signal Strength	Noise Level	Distance
Location A	-2dBm	-96dBm	2m
Location B	-91dBm	-96dBm	30m

For the measurements we used the `ttcp` benchmark tool [10], which sends a number of packets of a specified size to a receiver using TCP (or UDP), reporting at the end various transfer related metrics. We also used the `tcpdump` tool [11] to record detailed logs of all packets sent and received by the wireless interfaces during each test. Logs were later processed by `tcptrace` [12], an analysis tool that provides statistical and graphical analysis of TCP/IP traffic, correlating incoming and outgoing packets in order to compute performance statistics at both the IP and TCP layers. These statistics include estimated congestion window size, number of out-of-sequence and duplicated segments, throughput and RTT.

C. Measurement Procedure

We present below measurements with the clients in two different locations, while the AP is fixed. Location A is close to the AP and location B is far from the AP. Table III shows the signal strength, noise level and physical distance of each location from the AP. Channel noise was nearly constant during all tests, indicating that no sources of interference were present, while the signal strength experienced some deviation from time to time, due to the movement of people in the room.

The measurements consisted of executing `ttcp` with appropriate parameters in order to send 15MB of data; each run was monitored with `tcpdump`. The *Maximum Segment Size* (MSS) used by TCP was 1460 bytes, that is, the maximum LAN packet size minus 40 bytes for headers. The main test parameters were transfer direction (uplink/downlink), RTS (On/Off), and the *signal to noise ratio* (SNR) defined by the location of the clients. A test script automating the above procedure repeated each test five times, allowing us to estimate the variance. The script recorded `ttcp` output and SNR levels at both endpoints of the transfer. The `tcpdump` output files were used with `tcptrace` to generate dynamic TCP behaviour data that were plotted using `xplot` [13]. Among the possible measurements produced by `tcptrace`, we present in this paper the following:

- *Throughput*: It is computed by dividing the actual bytes transferred by the transfer time. This was used to double-check the throughput estimation of `ttcp`.
- *Congestion window*: As there is no direct way to determine the TCP congestion window without instrumenting the TCP code at the sender, the amount of outstanding (unacknowledged) data was used to estimate the congestion window size. The `tcptrace` tool measures the maximum, minimum, average and weighted average values for outstanding connection data.

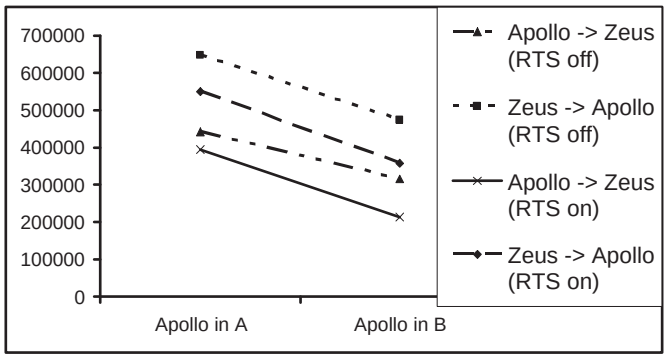

Fig. 1. Average uplink/downlink throughput (bytes/sec) (single TCP connection.

III. Measurement Scenarios and Analysis

A. Single Connection Measurements

In this section we present measurements involving a single TCP connection between Apollo and Zeus. After preliminary tests in many locations, we decided to present results from locations A and B, as these were representative of the results seen in other locations. For each location, we performed tests in both directions (uplink/downlink) for two different RTS threshold values: i) 2346 bytes, which is larger than the segment size of the sender, thus disabling the RTS/CTS mechanism, and, ii) 200 bytes, which is smaller than the segment size of the sender but larger than the size of a TCP acknowledgment segment, thus enabling RTS/CTS for data but not for TCP level acknowledgments.

The following observations can be made from these tests:

1) Throughput values are substantially lower than the available bandwidth (about 50% at best). Some explanations for this behaviour are given in [2], [3], but they are not within the scope of this paper.
2) Uplink and downlink throughput differ. The throughput achieved when Zeus was the TCP sender (downlink) was higher in all cases (Figure 1). This asymmetry was expected due to the differences between client and AP hardware. The AP's (Zeus) radio is more powerful (during the experiments we configured it at 100mW) and is also equipped with two 3dbi antennas working in diversity mode. Thus, we had better reception when Zeus was the sender, verified by the fact that the received SNR at Apollo was higher than it was at Zeus.
3) Enabling the RTS/CTS mechanism has a negative effect on achieved throughput (Figure 1). This was also expected since the RTS/CTS mechanism imposes overhead for each frame sent, by first exchanging an RTS/CTS frame pair, leading to decreased TCP throughput.
4) Lower signal strength leads to a greater frequency of changes in the estimated TCP congestion window. The result is a more "jagged" diagram for the estimated congestion window at location B (low signal strength), as opposed to a smoother congestion window diagram at location A (high signal strength) (Figure 2).

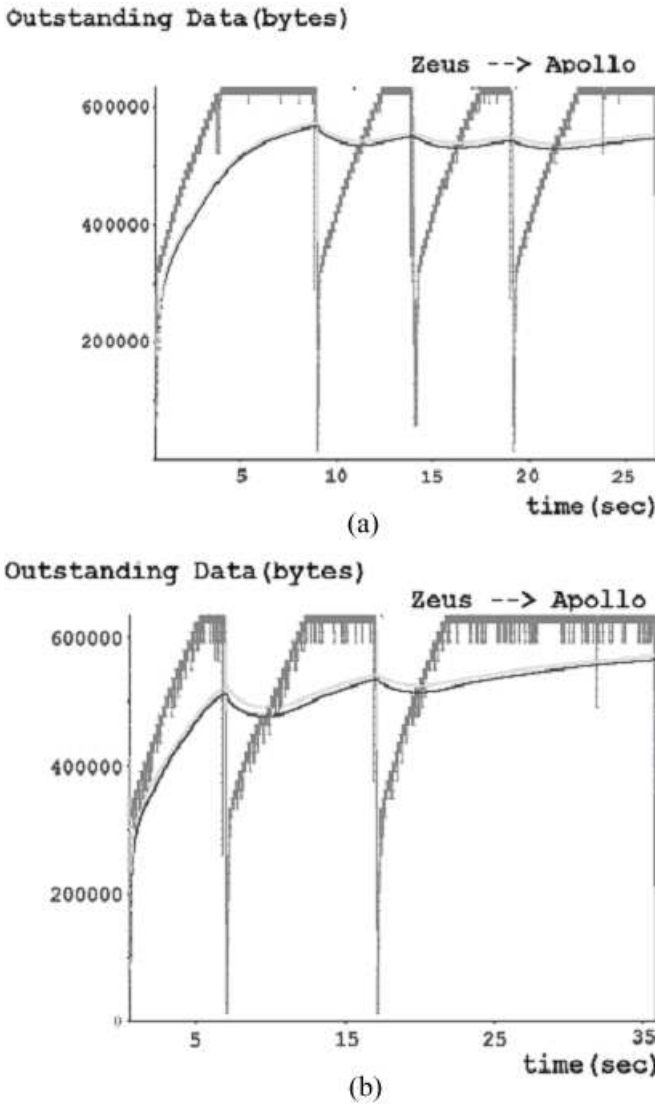

Fig. 2. Outstanding data (estimated congestion window size) at (a) location A and (b) location B for downlink traffic (RTS/CTS disabled, single TCP connection).

B. Multiple Connection Measurements

Having established above a performance baseline from the single connection measurements, we then added a second wireless client. Our goal was to investigate how the achieved TCP throughput was distributed between two competing TCP connections running on different hosts sharing the same wireless channel. We only used a single TCP connection per host since we were interested in the total throughput achieved by each host, not in the manner that throughput is distributed between competing TCP connections on the same host.

In the first scenario that we examined, the signal strength between the laptops and the AP was roughly the same and equal to its value in location A. This was achieved by arranging the hosts in a triangular topology where both clients were about the same distance from the AP and one another. Again, we ran tests in both the uplink and downlink directions, with the RTS/CTS mechanism either enabled or disabled.

As the left side of Figure 3 shows, in this scenario total throughput was divided evenly among the two connections in both the uplink and downlink directions, whether RTS/CTS was enabled or not. This is particularly important in the uplink scenario, where the two clients are directly competing for use of the medium, unlike in the downlink scenario where the AP alone is sending data to both clients.

Another observation is that the total throughput achieved is greater with two competing clients than with one client. This is due to the conservative transmission behaviour of TCP: it is easier to get two TCP connections to send with rate R simultaneously than to achieve a rate of $2R$ with a single TCP connection. These results indicate that the IEEE 802.11 MAC protocol (CSMA/CA) is dividing throughput "fairly" when the signal quality is equal.

In addition, the observations that we made for the single connection measurements, that is, that an asymmetry exists between the uplink and downlink throughput and that the throughput is decreased when RTS/CTS is enabled, still apply. The RTS/CTS mechanism is unnecessary in this scenario, since the two clients can detect each other's transmissions.

In order to investigate what happens when the competing clients experience different signal quality levels, in the second scenario Apollo was moved to location B, while Ares remained at location A, so that Apollo would experience worse signal level conditions than Ares. Throughput results from this scenario are shown on the right side of Figure 3.

In the downlink direction, with RTS/CTS disabled, in the single connection tests Apollo achieved a throughput of 474,680 bytes/sec at location B (Figure 1), while with Zeus sending data to both Ares in location A and Apollo in location B, the throughput at location B decreased to 196,550 bytes/sec (Figure 3). Similarly, in the single connection tests Apollo achieved a throughput of 647,822 bytes/sec at location A (Figure 1), while with Zeus sending data to both Ares at location A and Apollo at location B, the throughput in location A decreased to 449,053 bytes/sec (Figure 3). We observe that the host at location A and the host at location B lost comparable amounts of bandwidth when competing against each other, as opposed to when operating in isolation: the throughput reduction in location A was 278,310 bytes/sec while at location B it was 198,769 bytes/sec.

The throughput reduction is however considerably different in relative terms: the host at location B lost 58.59% of its throughput, while the host at location A lost only 30.68% of its throughput. By comparing the two sides of Figure 3, we can see the reason: in the competing clients scenario the throughput of the host at location A (Ares) was considerably increased when its competitor was moved to location B (Apollo). That is, the TCP connection from Zeus to Ares took advantage of the decreased transmission rate of the connection from Zeus to Apollo, thus achieving higher throughput.

This unfair TCP throughput allocation is due to the delays incurred by MAC layer retransmissions of corrupted frames when Apollo is in location B, where the signal strength is lower and the frame error rate is higher. These delays are attributed to congestion by TCP, thus lowering the transmission rate of this connection and allowing the competing

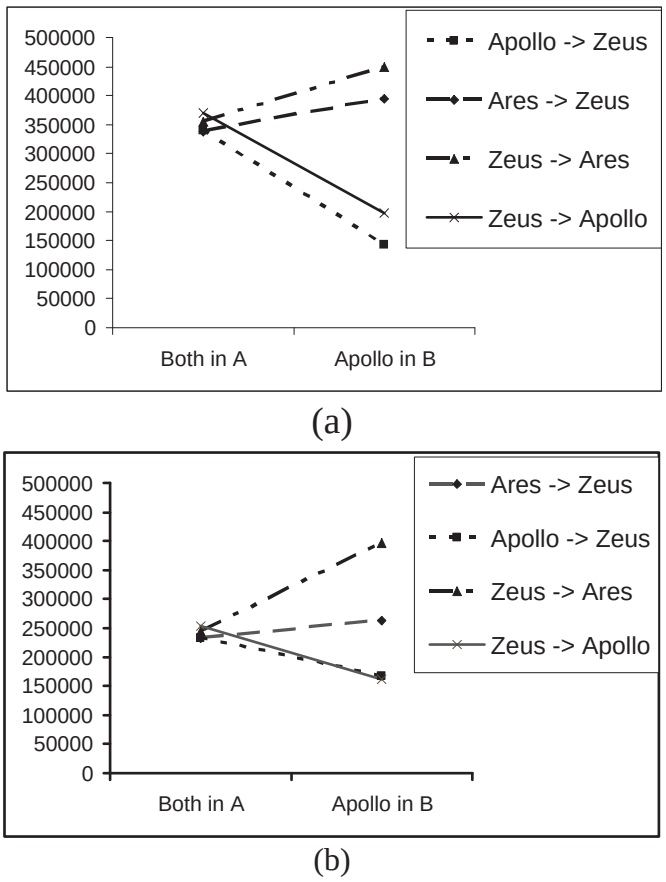

Fig. 3. Average uplink/downlink throughput (bytes/sec) with (a) RTS/CTS disabled and (b) RTS/CTS enabled (two TCP connections).

TCP connection to grasp a larger share of the bandwidth. As a result, the competing connection ends up with higher throughput than when both hosts had equal signal strength. This is evident by looking at the estimated congestion window sizes of Ares and Apollo in this scenario (Figure 4): Ares stabilizes its estimated congestion window at a large value, while Apollo keeps opening and closing it.

Enabling the RTS/CTS mechanism in the downlink direction decreased all throughput values, as expected. However, the overall behaviour followed the pattern observed with RTS/CTS disabled: when one host was moved to location B, the host at location A improved its performance at the expense of the host at B. This is reasonable, as in the downlink case the only data sender is Zeus, hence no conflicts occur and the sole impact of RTS/CTS is its overhead. In the reverse direction where there is contention, only acknowledgments are transmitted, which are too small to use RTS/CTS.

It was only in the uplink direction of this scenario that we expected the RTS/CTS mechanism to have some impact, since the two clients competing for the wireless medium were so positioned as to not be able to detect each other's transmissions all the time. We verified that this was the case by testing the link between Ares in location A and Apollo at location B and finding connectivity to be unstable. Indeed, in the uplink direction, with RTS/CTS enabled, the

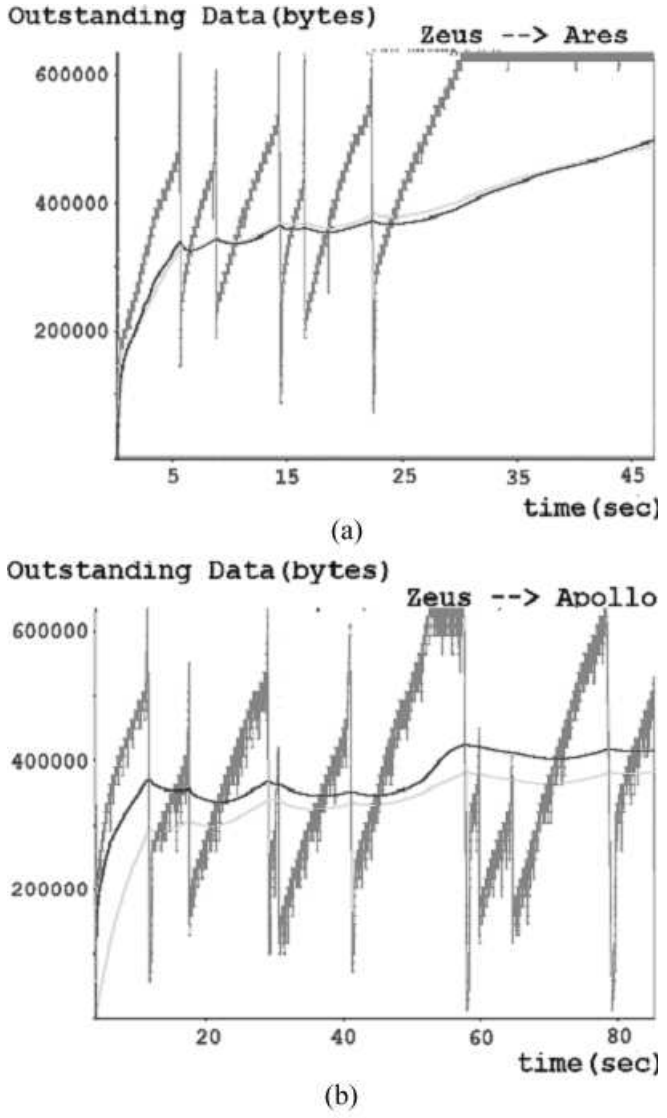

Fig. 4. Outstanding data (estimated window size) for downlink traffic (a) from Zeus to Ares and (b) from Zeus to Apollo (RTS/CTS disabled, two TCP connections, Ares in location A and Apollo in location B).

gap between the performance of the host at location A and the host at location B was considerably smaller (Figure 3). That is, the unfairness factor was smaller with RTS/CTS enabled, even though the total throughput achieved was worse: the ratio between Apollo's and Ares' throughput was 0.635 with RTS/CTS enabled, compared to 0.36 with RTS/CTS disabled. It seems then that in this case the RTS/CTS mechanism was beneficial with respect to the fairness of TCP throughput sharing between the two hosts at location A and location B.

A possible explanation for this phenomenon is that the RTS/CTS mechanism, by introducing delays in the TCP transmissions and throttling back the TCP sender, makes the client at location A less aggressive when competing for throughput with the client at location B, thus preventing the client with the higher signal strength from grasping most of the available bandwidth. Apollo's throughput (Location B) may have remained low, but the decrease was smaller. Without RTS/CTS Apollo's throughput was reduced from 341,243 bytes/sec to 171,225 bytes/sec (49.82% decrease), while with RTS/CTS it was reduced from 232,567 bytes/sec to 166,615 bytes/sec (28.35% decrease). That said, the overall effect of enabling the RTS/CTS mechanism was clearly a negative one for both the individual and total throughputs achieved.

IV. Some Measurement Issues

Before we conclude our analysis, we discuss some measurement difficulties that we encountered and which we believe should be taken into account in similar tests.

Changes in signal strength levels can have great impact on TCP throughput. When the corridor between the AP and location B became crowded with people, for example, during class breaks, throughput dropped dramatically.

When performing measurements from locations with low signal levels, the results are more susceptible to random disturbances. In order to collect a valid set of results, such tests should be repeated a significant number of times.

V. Conclusions and Future Work

Our study indicates that there exist serious unfairness issues in the distribution of WiFi bandwidth among multiple hosts with different signal strengths when TCP is used. It is not trivial to predict how this affects the QoS policies applied in such an environment. As we observed in our experiments, when both clients were downloading data at the same time, the performance of the client with the lower signal quality was disproportionately affected by the client with the higher signal quality. By employing a QoS policy on the AP that enforces an upper bound on the rate of data sent to each client, for example, the throughput that each client would achieve if all clients had equal signal strengths, it is possible that clients with lower quality signals would not be penalized as much.

Self-organized wireless systems providing roaming between peers (e.g., see [14], [15]) may use QoS policies to assign bandwidth shares to users in proportion to their contribution. Our measurements indicate however that, despite nominally different bandwidth allocations, users with lower contributions may get better treatment due to their advantageous position. An issue for further study is how signal strength could be taken into account so as to retain the reciprocity motive.

Another issue for further study is the role of the RTS/CTS mechanism with respect to fairness. The scenarios that we studied show that this mechanism considerably reduces total TCP throughput achieved. On the other hand, measurements in scenarios where clients are greater in number and experience diverse signal quality levels indicate that the RTS/CTS mechanism can have a positive effect on the fairness of TCP throughput sharing. It is an open issue whether it is possible to obtain these benefits of RTS/CTS without its penalties.

Finally, we plan to extend our measurements to include:

1) Multirate tests in order to check the impact that the bit rate adaptation algorithm has on the transport layer.
2) UDP and/or more TCP implementation tests, in order to evaluate their performance with respect to fairness and correlate our results with the MAC layer.

ACKNOWLEDGEMENTS

We would like to thank Elias C. Efstathiou for his valuable assistance at various stages of this work.

REFERENCES

[1] The tc tool, available at: http://developer.osdl.org/dev/iproute2/
[2] *TCP Performance Issues over Wireless Links*, G. Xylomenos, G.C. Polyzos, P. Mähönen, and M. Saaranen, *IEEE Communications Magazine*, vol. 39, no. 4, 2001, pp. 52-58.
[3] *TCP & UDP Performance over a Wireless LAN*, G. Xylomenos and G.C. Polyzos, *IEEE INFOCOM 1999*, pp. 439-446.
[4] *Wide-Area Internet Traffic Patterns and Characteristics*, K. Thompson, G.J. Miller, and R. Wilder, *IEEE Network*, vol.11, no. 6, 1997.
[5] *Longitudinal study of Internet traffic in 1998-2003*, M. Fomenkov, K. Keys, D. Moore, and K. Claffy, *Winter International Symposium on Information and Communication Technologies (WISICT)*, January 2004.
[6] The mii-tool, available at: http://linuxip.net/html/tools-mii-tool.html
[7] *Performance Analysis under Finite Load and Improvements for Multirate 802.11*, G.R. Cantieni, Q. Ni, C. Barakat, and T. Turletti, *Computer Communications*, vol. 28, no.10, pp. 1095-1109, 2005.
[8] *Performance Anomaly of 802.11b*, M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, *IEEE INFOCOM 2003*, March 2003.
[9] Debian GNU/Linux, available at: http://www.debian.org/
[10] The ttcp tool, available at: http://ftp.arl.mil/ftp/pub/ttcp/
[11] The tcpdump and libpcap tools, available at: http://www.tcpdump.org/
[12] The tcptrace tool, available at: http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html
[13] The xplot tool, available at: http://www.xplot.org/
[14] *Self-Organized Peering of Wireless LAN Hotspots*, E.C. Efstathiou and G.C. Polyzos, *European Transactions on Telecommunications*, vol. 16, no. 5, September/October 2005.
[15] *Reducing Management Complexity through Pure Exchange Economies: A Prototype System for Next Generation Wireless/Mobile Network Operators*, P.A. Frangoudis, E.C. Efstathiou, and G.C. Polyzos, *12th Annual Workshop of the HP OpenView University Association (HP-OVUA)*, Porto, Portugal, July 2005.