A Smart Access Point Solution for Heterogeneous Flows

Claudio E. Palazzi, Nicola Stievano
Dipartimento di Matematica Pura ed Applicata
Universitá di Padova
Via Trieste 63, 35121 Padova, Italy
Email: cpalazzi@math.unipd.it
nstievan@studenti.math.unipd.it

Marco Roccetti
Dipartimento di Scienze dell'Informazione
Universitá di Bologna
Mura Anteo Zamboni 7, 40127 Bologna, Italy
Email: roccetti@cs.unibo.it

***Abstract*—Computer-centered services and broadband wireless connectivity are enabling the delivery of multimedia-based entertainment from the Internet to in-house wireless devices and appliances. In this context, rich-media applications can be supported by different protocols that must share the same channel without affecting each other performances. Instead, with current systems, real-time applications (e.g., video streaming, online games) suffer from delays caused by the interference with elastic (e.g., TCP-based downloading sessions) ones. In addition, elastic applications may also unfairly damage each other when featured with different round-trip times (RTTs) between clients and servers, even if sharing the same bottleneck. In this article we provide insight on these problems and show how a solution based on a smart access point may actually solve them, allowing a fair coexistence between heterogeneous flows, even when featured with different transport protocols and different RTTs.**

***Keywords*-Computer-Centered Home Entertainment, Home Network, Smart Access Point, Wireless Multimedia.**

I. Introduction

The home information system is changing, more and more devices present at home such as personal computers, consoles and television may make be connected with each other or to the Internet through wireless connectivity. Indeed a possible scenario is that of a home gateway providing wireless connectivity to the various devices in a house through an access point (AP), in order to enable file downloading, Internet browsing, movie streaming, video-chatting, online gaming and many other connectivity-based multimedia applications [1], [2]. Yet not much work has been done to understand whether the Internet protocols will be able to efficiently support this complex scenario. It is expected that the transport protocols utilized by applications will remain those that have been in use for the last 30 years: the Transmission Control Protocol (TCP) for elastic (e.g., downloading) applications and the User Datagram Protocol (UDP) for real-time ones (e.g., video/music streaming, online gaming) [3].

In a home entertainment scenario new problems emerge, requiring the employment of specific solutions. For instance, the TCP's probing mechanism to utilize more and more bandwidth and the presence of large buffers at the bottleneck of a connection have been demonstrated to negatively affect real-time applications as they increase the per-packet delivery latency [4], [5]. Moreover, not only are TCP-based elastic applications harmful towards real time ones, but they can also harm each other. For instance, when two devices inside a house are engaged in TCP sessions to download some big (multimedia) file through the same bottleneck, one would expect that they achieved a fair sharing of the available bandwidth; instead, small round-trip time (RTT) downloading sessions unfairly capture most of the bandwidth with respect to longer RTT downloads [6], [7].

In order to enable an efficient home network to support simultaneous and heterogeneous multimedia applications, we need to achieve three goals: i) low per-packet delays, ii) high downloading throughput, and iii) fair utilization of the shared bottleneck. To this aim, we discuss here a solution that makes use of a smart AP, standard protocol features, and available information on the on-going traffic; through simulative experiments we demonstrate how hour solution, named *Smart Access Point with Low Advertised Window (SAP-LAW)*, is able to achieve the three goals mentioned above.

The rest of the paper is organized as follows. Section II discusses the SAP-LAW protocol and the solution developed for heterogeneous RTT flows. The experimental testbed is presented in Section III, whereas obtained results are shown in Section IV. Finally, Section V concludes this paper.

II. An RTT-fair SAP-LAW

One of the main features of TCP is represented by its congestion control functionality by which the sending rate of a transmission session continuously grows, queuing packets on the buffer associated with the bottleneck of a connection, until some packet gets lost; at that point the sending rate is halved before starting again its growing cycle. If one considers that the same wireless connection might be shared by several devices and applications, it becomes evident how this aggressive behavior of TCP-based connections (i.e., elastic downloading applications) causes packet buffering that delays all packets passing through that bottleneck, even those belonging to real-time applications [4].

This problem can be solved through the use of a smart AP that monitors all the on-going traffic and forces TCP flows to not exceed a certain amount of bandwidth consumption [5]. This can be done through a regular feature of standard TCP protocol: the advertised window. In essence, an appropriate advertised window is computed for the TCP flows so as to maximize the throughput of elastic applications but, at the same time, to not generate queues at the bottleneck that would jeopardize the requirements for low per-packet delays of real-time applications. This computed value for the advertised window is then included in all the TCP's acknowledgments (ACKs) traveling through the AP by the AP itself. The formula utilized to compute the appropriate value for the advertised window of TCP-based elastic flows can simply be as follows:

$$maxTCPrate(t) = \frac{(C - UDPtraffic(t))}{\#TCPflows(t)} \quad (1)$$

where $UDPtraffic$ is the amount of bandwidth occupied by the UDP-based traffic at time t, $\#TCPflows$ is the concurrent number of TCP flows at time t, and C is the capacity of the bottleneck link.

However, as it is evident, (1) does not solve completely the well known RTT bias that affects TCP-based flows. In essence, since the dependence from the RTT in determining the ratio of the returning ACKs and hence the increase of the sending rate, TCP-based flows with small RTT unfairly capture the available bandwidth with respect to other flows [6]. When leveling the sending windows through (1), the throughput of each TCP-based flows becomes equal to the ratio between the constant sending window (at steady state equal to the computed advertised window) and the flow's RTT, thus maintaining some RTT unfairness among the various flows. The outcome is better than with regular TCP where, in addition, the sending window of small RTT flows always results bigger than the sending window of bigger RTT flows; yet, the RTT unfairness problem is not completely solved.

As an example, Fig. 1 shows the average throughput of each of three simultaneous downloads of a multimedia file from three different servers. The three servers are located at 30, 50, and 100 ms, respectively, of RTT from three downloading clients inside the house. This simple but revealing example considers a regular AP implementing IEEE 802.11g MAC layer and the last mile wireless link between the AP and the clients within the house represent the shared bottleneck of the connections (*circa* 19 Mb/s of bandwidth). The chart clearly shows the unfair divergence among the data rates.

From these considerations arises the need to modify (1) in order to have different maximum transmission rates, through different advertised windows, for flows with different RTTs. The new formula for a flow i can be written as follows:

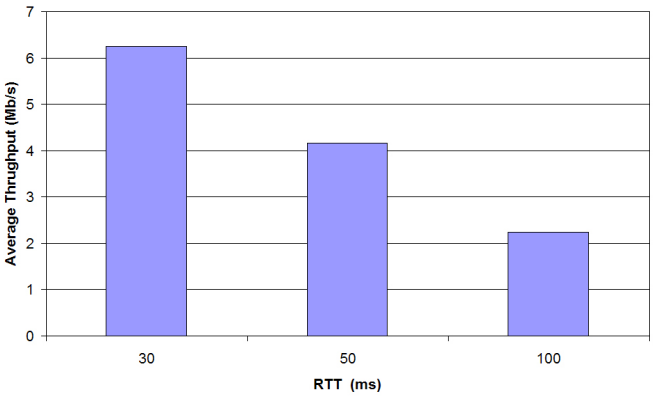

Figure 1. Average thrughput for three simultaneous flows with different RTTs (30, 50 and 100 ms, respectively).

$$maxTCPrate_i(t) = \frac{(C - UDPtraffic(t)) * RTT_i}{(a_1 + a_2 + a_3)} \quad (2)$$

$$a_1 = \frac{RTT_1}{avg.RTT_{min}}, a_2 = \frac{RTT_2}{avg.RTT_{min}}, a_3 = \frac{RTT_3}{avg.RTT_{min}},$$

where RTT_1, RTT_2, and RTT_3 are the smallest RTTs for the three TCP-based flows sharing the AP, and $avg.RTT_{min}$ is the average among these smallest RTTs.

The rationale behind (2) is the fact the throughput at steady state can be computed as the ratio between the maximum sending rate and the RTT for that connection. Therefore, if two connections have different RTTs but we want equal throughput, then their maximum sending rates should be simply chosen with an inverse proportion.

III. Simulation Assessment

We want to demonstrate the efficacy of our solution in ensuring the three goals listed in Section I:

- low per-packet delays;
- high downloading throughput;
- fair utilization of the shared bottleneck.

To this aim, we have built an experimental scenario utilizing the well-known NS-2 simulator with modifications to implement our SAP-LAW solution [8]. In particular, we intend to analyze a general house environment with four devices connected to the Internet through the wireless connectivity provided by an AP. The main settings and parameters of the experimental scenario are as follows.

The distance between each device and the AP is 10 m and the MAC layer parameters have been set accordingly to the IEEE 802.11g standard allowing us to reach a maximum wireless bandwidth of *circa* 19 Mb/s; this represents a reasonable maximum value for TCP-based flows over the declared 54 Mb/s, even in the real world [9].

For the wireless medium in our simulations we have utilized the *Shadowing Model* present in the NS-2 simulator

which realistically simulates signal fading. We followed the directions provided by the official NS-2 manual to represent a home environment partitioned into several rooms. Specifically, the *path loss exponent* and the *shadowing deviation* parameters have been set equal to the worst possible case suggested for an indoor environment: 4 and 9, respectively.

Our experimental campaign can be divided into two main parts. In the first one, we have evaluated the efficacy of SAP-LAW when employed in a context with heterogeneous applications, i.e., TCP-based elastic applications (FTP download sessions) and UDP-based real-time applications (videostream, online gaming, video-chat). Instead, in the second one, we have focused on the simultaneous presence of TCP-based flows with heterogeneous RTTs. The following two subsections describe the two simulative scenarios.

A. Simulative configuration #1: Coexistence evaluation

In the first set of simulations we consider four devices running different applications supported by either TCP or UDP as transport protocol. The four applications experience different RTTs and start at different times (see table I). In order to uplift the trustworthiness degree of the simulations, we exploit real trace files for the videostream and for the video-chat applications. Specifically, the application trace files correspond to a high quality MPEG4 version of *Star Wars IV* for the movie, and to two VBR H.263 Room-Cams for the video-chat, as available in [10]. Parameters characterizing the game-generated traffic are chosen following directions provided by scientific literature related to this field [11].

We assume the user in the house engaged in one of the very popular first person shooter games, e.g. Quake or Counter Strike, with other $\tilde{2}5$ remote players connected through the Internet. To model the packet size and the inter-arrival time of the traffic generated by the online game, we exploit the model suggested in [11], which is based on real game platform measurements. Game events are hence generated at client side every 60 ms, whereas the server transmits back game state updates every 50 ms toward the client. Moreover, game packet sizes generated by the client inside the house and the server are set to 42 bytes and 200 bytes, respectively.

In this context, we study the performance achieved in terms of per-packet delays and total throughput achieved when changing the size in Mb/s of the link between the AP and the Internet (the link between AP and W0 in Fig. 2): 4, 10, and 20 Mb/s. When the bandwidth available on this link is less than or equal to 19 Mb/s then it also represents the bottleneck of the network. Otherwise, the bottleneck is located at the wireless link between the AP and the wireless nodes; as said, this oscillates around 19 Mb/s.

Table I
SIMULATIVE CONFIGURATION #1

Flow Type	Protocol	From	To	Start	End	RTT
Video-Stream	UDP	N4	W0	0 s	180 s	0 ms
Online Game	UDP	N1	W1	45 s	180 s	40 ms
Video-Chat	UDP	N3	W3	90 s	180 s	60 ms
FTP	TCP	N2	W2	135 s	180 s	80 ms

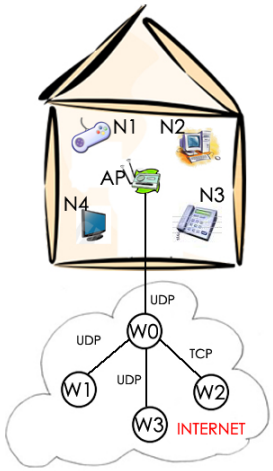

Figure 2. Scenario of the simulative configuration #1.

Table II
SIMULATIVE CONFIGURATION #2

Flow Type	Protocol	From	To	Start	End	RTT
FTP 1	TCP	N2	W2	1 s	180 s	30 ms
FTP 2	TCP	N3	W3	1 s	180 s	50 ms
FTP 3	TCP	N4	W4	1 s	180 s	100 ms
Online Game	UDP	N1	W1	45 s	180 s	20 ms

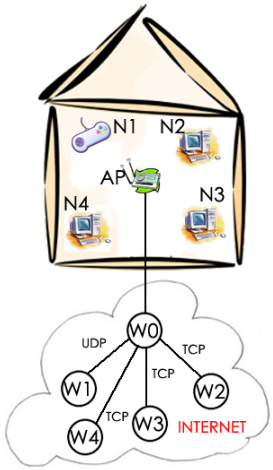

Figure 3. Scenario of the simulative configuration #2.

B. Simulative configuration #2: RTT fairness evaluation

In the second set of simulations, we consider a single UDP flow generated by the online game applications (with

the same characteristics described above for the simulative configuration #1) and three TCP flows generated by three FTP applications. The RTT values, the start time, and the protocols employed are summarized in table II. The scenario for this set of simulations is depicted in Fig. 3.

IV. Experimental Results

In this section we present results collected through the two considered simulative scenarios. In particular, to evaluate the capability to support both elastic (even with different RTTs) and real-time flows, we show and discuss the shape of the TCP sending window, the throughput achieved, and the inter-arrival time of online game packets.

A. Coexistence evaluation

In the first set of simulations (simulative configuration #1), we have studied the behavior of TCP flows with respect to different capacities of the bottleneck link.

By observing Fig. 4 it is evident that if we have the bottleneck (i.e., the link between W0 and AP in Fig. 2) with values of 4 Mb/s, the sending window (*swnd* in the chart) remains below the pipe size of the connection (*RTTxBW* in the chart). This is due to the ability of SAP-LAW in maintaining the TCP sending rate just below the available bandwidth diminished by the portion of the channel consumed by UDP-based applications.

Coherently, in Fig. 5, we see that even with a bottleneck value of 10 Mb/s, the sending window is limited by our SAP-LAW solution which smoothes its behavior avoiding the classic TCP's saw-tooth shape.

Finally, Fig. 6 shows the sending window for the TCP flow, when the link between AP and W0 (see Fig. 2) has a capacity of 20 Mb/s. In this case, the actual bottleneck of the connection is represented by the wireless link between the AP and the wireless nodes in the house. The capacity of this link is *circa* 19 Mb/s: its actual capacity oscillates depending on wireless channel fluctuations. Obviously, if setting the link between AP and W0 with any value higher than 19 Mb/s, the chart would result very similar to Fig. 6 as the actual bottleneck (the wireless link) does not change.

Regardless of the bottleneck value in the considered scenario, SAP-LAW is able to reduce the queue delay at the bottleneck, speeding up the per-packet delivery delay. To demonstrate this, we consider the case with the bottleneck of 19 Mb/s located at the wireless link. In this configuration, Fig. 7 and Fig. 8 present the game packets's inter-arrival time without or with SAP-LAW, respectively: the wider the oscillations of the inter-arrival time, the higher the queuing delay suffered in average by game packets transiting through the link.

As it is evident, SAP-LAW is able to systematically and significantly reduce the inter-arrival time among packets as it eliminates queuing delays from the bottleneck link. Indeed,

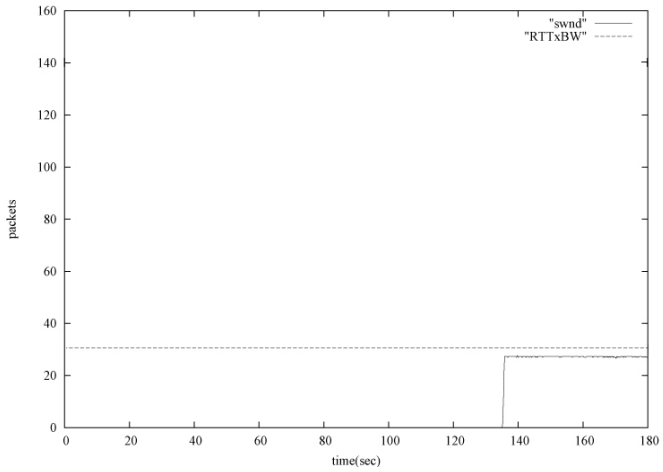

Figure 4. SAP-LAW sending window when employing (1): capacity of the link between W0 and AP (bottleneck) = 4 Mb/s.

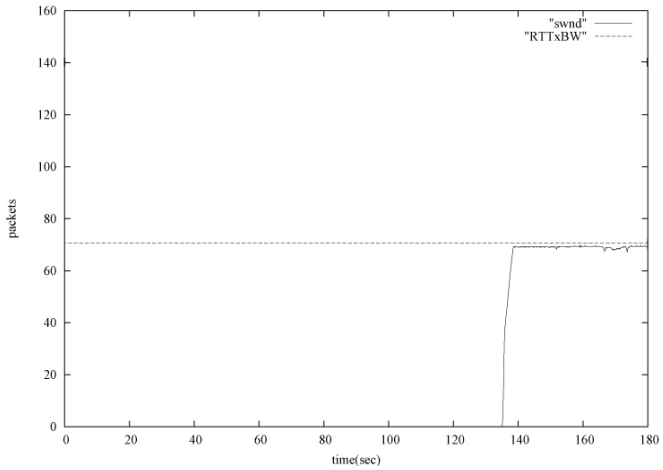

Figure 5. SAP-LAW sending window when employing (1): capacity of the link between W0 and AP (bottleneck) = 10 Mb/s.

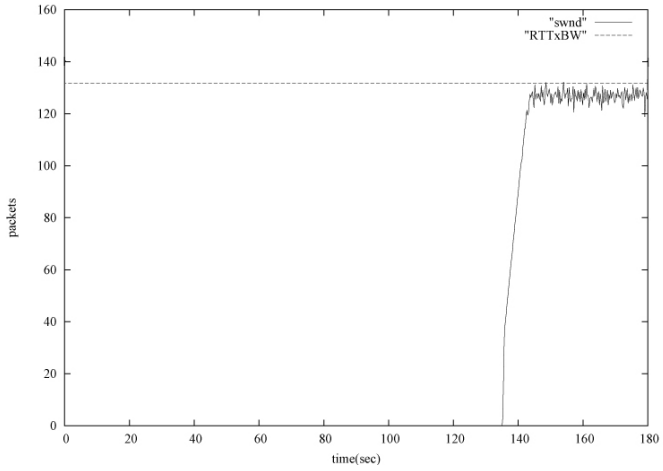

Figure 6. SAP-LAW sending window when employing (1): capacity of the link between W0 and AP = 20 Mb/s; the bottleneck is the wireless link among the AP and the wireless devices: *circa* 19 Mb/s.

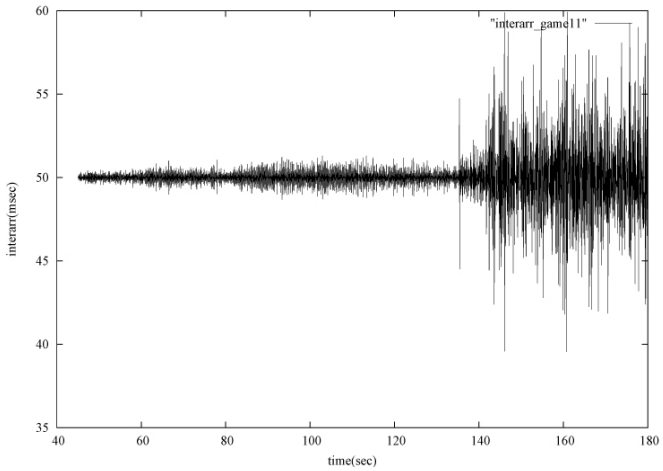

Figure 7. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. From 135 s, a FTP/TCP New Reno flow is competing for the channel.

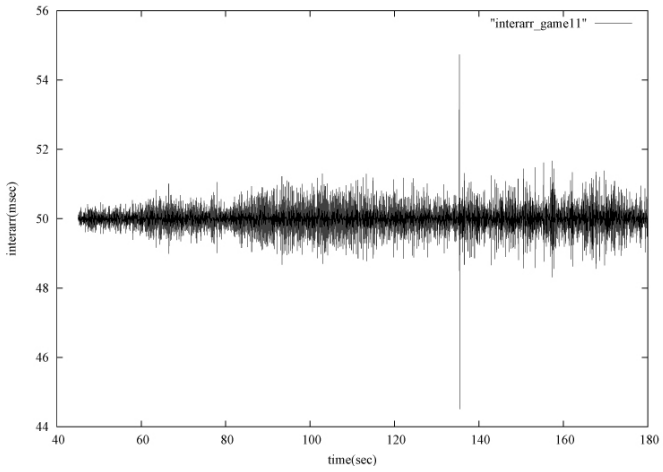

Figure 8. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. SAP-LAW with (1) and regular IEEE 802.11g are employed; from 135 s, a FTP/TCP New Reno flow is competing for the channel.

in Fig. 8 it is visible a single, and very limited, oscillation peak when the FTP/TCP session starts (at 135 s).

However, even if solving the bottleneck queuing problem, quickly delivering all game packets toward destination, yet SAP-LAW employing (1) does not guarantee RTT-fairness when heterogeneous FTP/TCP flows share the same bottleneck. In the next subsection we demonstrate this statement, also discussing (2) as a solution.

B. RTT fairness evaluation

In the second set of simulations (simulative configuration #2) we focus on the RTT-fairness problem and, to this aim, we observe the behavior of three TCP flows in the home scenario. First, we consider flows characterized by the same

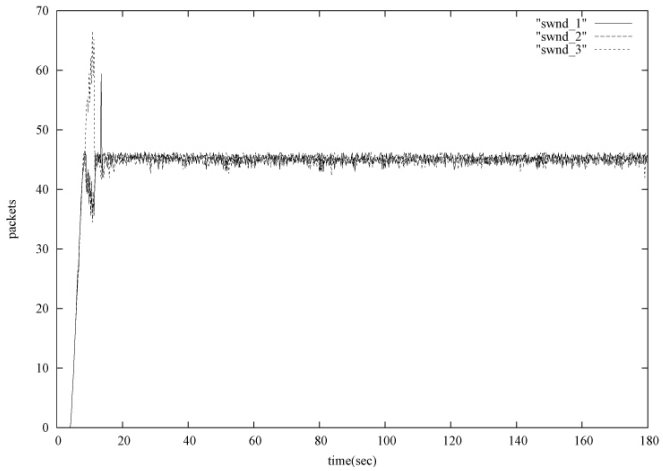

Figure 9. SAP-LAW sending window of three TCP flows with equal RTTs when employing (1).

RTT and then we study what happens when the RTTs are different.

In Fig. 9 we consider three TCP flows with equal RTT values and with SAP-LAW implementing (1). Under these conditions, we can notice that the three lines representing the sending windows have the same trend. Moreover, in Fig. 10 we can see that also the returning ACKs rates, and hence the throughputs, are the same for the three flows.

Instead, when the RTTs are different, the system will suffer by the RTT-unfairness problem discussed before. In particular, we consider now the case where three different FTP/TCP flows with 30, 50, and 100 ms of RTT, respectively, and the wireless link as the bottleneck (*circa* 19 Mb/s). As Fig. 11 shows, even in this scenario, the maximum sending window of the three RTT-heterogeneous TCP flow remains the same, as they all utilize SAP-LAW with (1). However, when having the same sending window but different RTTs, the actual data transmission rates will vary inversely with the corresponding RTT duration, as demonstrated by Fig. 12 that reports on the different returning ACK rates for the three flows. In the chart, *ack1*, *ack2*, and *ack3*, corresponds to the returning ACKs for the TCP flows with 30, 50, and 100 ms of RTT, respectively.

This is precisely the problem that we intend to solve with the use of (2): while maintaining the ability of SAP-LAW in ensuring low per-packet delays to real-time multimedia applications, we also want to guarantee RTT-fairness to elastic (TCP-based) flows.

As a result of enhancing SAP-LAW with the utilization of (2) in place of (1), Fig. 13 shows that the trends of the sending windows of the three TCP flows becomes different; specifically, (2) privileges the disadvantaged flows with higher RTTs. The chart clearly shows that the maximum sending window reachable by each flow becomes directly proportional with the flow's RTT. In this way, the through-

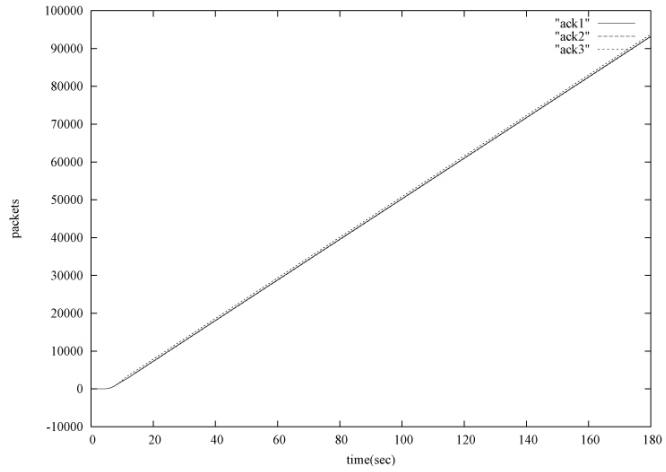

Figure 10. ACKs received from the sender of three TCP flows with equal RTTs when employing SAP-LAW with (1).

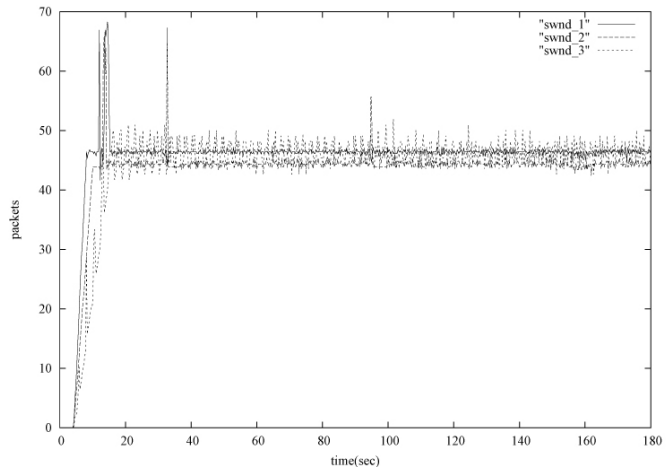

Figure 11. Sending window of three TCP flows with different RTTs when employing SAP-LAW with (1).

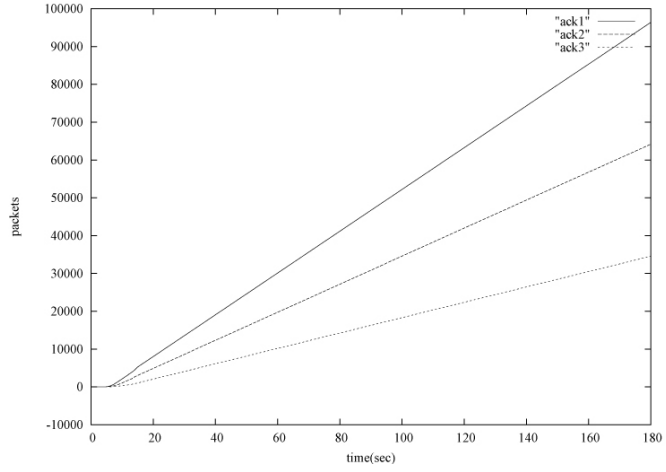

Figure 12. ACKs received from the senders of three TCP flows with different RTTs; SAP-LAW with (1) employed.

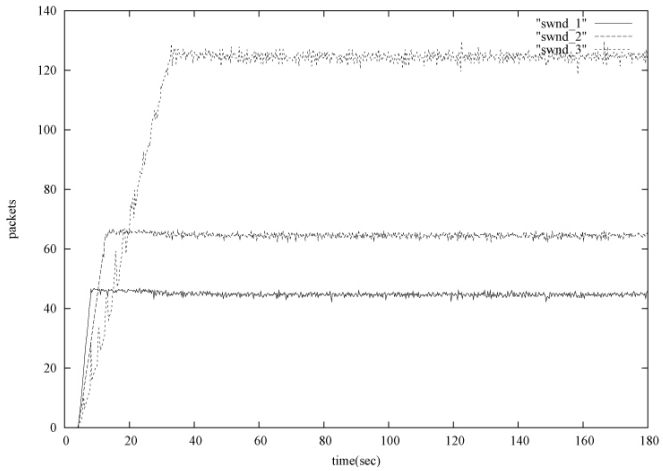

Figure 13. Sending window of three TCP flows with different RTTs when employing SAP-LAW with (2).

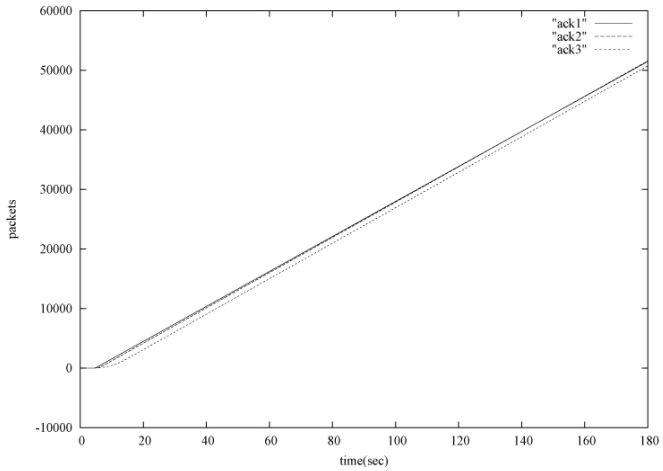

Figure 14. ACKs received from the sender of three TCP flows with different RTTs when employing SAP-LAW with (2).

puts are equalized as demonstrated by Fig. 14, which shows that the ACK rates of the three applications become almost identical.

Finally, having solved the RTT-unfairness problem, we want to be sure that our solution is able to preserve the other two properties mentioned in Section I: low per-packet delays and high downloading throughput. To this aim, Fig. 15 demonstrates that (2) is able preserve the first property, keeping a low variation of the inter-arrival time experienced by online game packets in the considered scenario. The values of the inter-arrival time is around 50 msec, corresponding to the inter-departing time, and achieving a very similar result as done with SAP-LAW when (1) was employed.

Moreover, Fig. 16 demonstrates the efficiency in terms of high downloading throughput. In fact, the chart shows that the total throughput achieved by the sum of the three considered flows reaches the maximum available on the

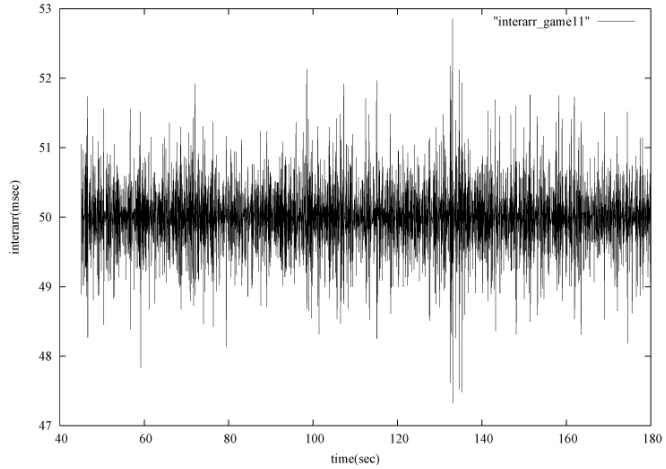

Figure 15. Measured inter-arrival time of online game packets when employing SAP-LAW with (2).

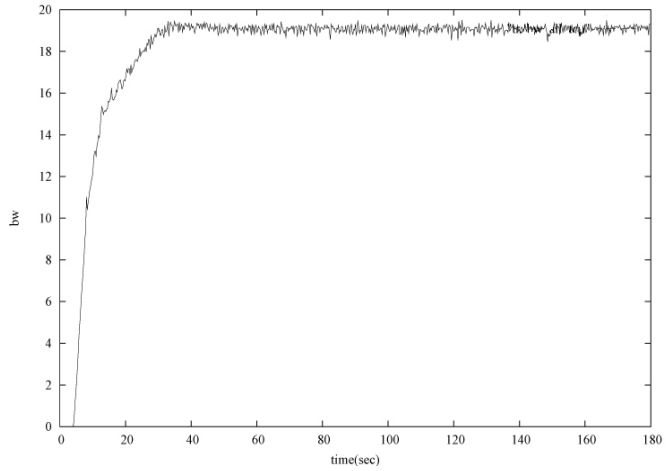

Figure 16. Throughput of the TCP flows when employing SAP-LAW with (2).

channel (*circa* 19 Mb/s) whereas in Fig. 1 we have seen that, employing (1) in the same network configuration, the maximum total throughput was less than 13 Mb/s.

V. Conclusion

In this article we have shown how to facilitate the coexistence among heterogeneous multimedia flows in a shared wireless communication channel. In particular, we aimed with success at providing three main goals:

- low per-packet delays;
- high downloading throughput;
- fair utilization of the shared bottleneck.

Our SAP-LAW solution is implemented through a very simple formula that can be easily introduced on real APs. The main result is that SAP-LAW facilitates the coexistence of heterogeneous multimedia applications in a complex environment such as a home where multiple applications may be simultaneously run by different persons.

The very encouraging results have convinced us in continuing with this work in two different directions: i) testing SAP-LAW in complex scenarios besides the home network and ii) the implementation of SAP-LAW on a real AP.

References

[1] M. Furini, "Mobile games: What to expect in the near future," in *in Proc. of GAMEON Conference on Simulation and AI in Computer Games*, Bologna, Italy, Nov 2007.

[2] A. Ploss, S. Wichmann, F. Glinka, and S. Gorlatch, "From a single- to multi-server online game: A quake 3 case study using rtf," in *of ACM Advances in Computer Entertainment (ACE 2008)*, Yokohama, Japan, Dec 2008.

[3] L. L. Peterson and B. S. Davie, *Computer Networks, a System Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996.

[4] C. E. Palazzi, G. Pau, M. Roccetti, and M. Gerla, "In-home online entertainment: Analyzing the impact of the wireless mac-transport protocols interference," in *in Proc. of IEEE International Conference on Wireless Networks, Communications and Mobile Computing (WIRELESSCOM 2005)*, Maui, HI, USA, Jun 2005, pp. 516– 521.

[5] C. E. Palazzi, S. Ferretti, M. Roccetti, G. Pau, and M. Gerla, "What's in that magic box? the home entertainment center's special protocol potion, reealed," *IEEE Transactions on Consumer Electronics, IEEE Consumer Electronics Society*, vol. 52, no. 4, pp. 1280–1288, Nov 2006.

[6] G. Marfia, C. E. Palazzi, G. Pau, M. Gerla, M. Y. Sanadidi, and M. Roccetti, "Tcp libra: Exploring rtt-fairness for tcp," in *in Proc. of the IFIP/TC6 NETWORKING 2007*, Atlanta, GA, USA, May 2007.

[7] C. Caini and R. Firrincieli, "Tcp hybla: A tcp enhancement for heterogeneous networks," *International Journal of Satellite Communications and Networking*, vol. 22, no. 5, p. 547566, 2004.

[8] "The network simulator, ns-2," http://www.isi.edu/nsnam/ns/.

[9] A. L. Wijesinha, Y. Song, M. Krishnan, V. Mathur, J. Ahn, and V. Shyamasundar, "Throughput measurement for udp traffic in an ieee 802.11g wlan," in *in Proc. of 6th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)*, Towson, MD, USA, May 2005.

[10] "Movie trace files," http://www-tkn.ee.tuberlin.de/research/trace/ltvt.html.

[11] J. Farber, "Traffic modelling for fast action network games," *Multimedia Tools and Applications*, vol. 23, pp. 31–46, 2004.