An End-to-End Technique to Estimate the Transmission Rate of an IEEE 802.11 WLAN

Antonio A. de A. Rocha Rosa M. M. Leão Edmundo de Souza e Silva
Federal University of Rio de Janeiro, Brazil
{arocha,rosam,edmundo}@land.ufrj.br

***Abstract*— The deployment of wireless LANs (WLANs) has been steadily increasing over the years and estimating the actual bit rate of a WLAN device is important for management and applications such as those that can adapt their transmission rates according to the network characteristics. We propose a simple and accurate active measurement technique to infer the bit rate of an IEEE802.11 device. The proposed method is based both on a recently proposed technique to infer the type of access network and on the packet pair approach, but adapted to take into account the overhead caused by the IEEE802.11 control and the existence of concurrent WLAN traffic. Furthermore, the technique does not require the WLAN to be the bottleneck link in the path from the measuring point to the end computer. Results from simulation and from measurements show that the approach is accurate to infer WLAN access point rates both in scenarios where the WLAN devices can adapt their rates as well as WLANs with fixed transmission rates.**

I. Introduction

Wireless local area networks (WLANs) based on the IEEE 802.11 standards [1] have become one of the most popular access networks. The high transmission rates offered by the current standards (802.11a/b/g) and the significant reduction in the costs of the equipment have greatly eased the deployment of this technology on public locations (e.g. airports, universities, libraries, cafes, shopping centers) and residences.

Access networks consist of the links between end systems and edge Internet routers. Examples of such links are dial-up lines, ADSL, Cable Modem, Ethernet and WLAN. The various types of access networks have different characteristics in terms of transmissions speeds, physical media, and capacity in the reverse and forward links.

Identifying the connection type of the last hop of an Internet path can be very useful for several applications, such as video streaming, overlay networks and peer to peer. For example, in order to improve performance, it may be advantageous to peer-to-peer applications to know the type of access network (as suggested in [2]) and the transmission rates of the clients, in order to choose the *best* peer to connect. As another example, there are proposals to improve TCP performance when the last hop is a WLAN [3] or a cable modem [4].

These examples explain why recent research was devoted to characterizing the type of connection available to an end user. The work of [5] was the first aiming at differentiating between wireless and wired network access. It is based on the observation of round trip times (RTT's) of TCP connections. In [2] the authors proposed a method to identify the type of an access network (Ethernet, WLAN, or low bandwidth).

The study presented in this paper builds on the works of [2] and [5]. We propose a technique to infer the **transmission rate** (capacity in bits per second) of the last hop of an Internet path, provided that one could establish that the last hop is a WLAN using the IEEE802.11 standard. Therefore, we rely on algorithms such as that in [2] to determine the connection type and, from that, our algorithm obtains the transmission rate. Our technique is an extension of the traditional packet pair method. The basic idea of our approach is to send two back-to-back packets with different sizes. From the dispersion between these packets at the receiver, one can obtain the desired measure.

Packet pair and packet train approaches have been extensively used to estimate the bottleneck link capacity or available bandwidth of an Internet path [6], [7], [8], [9], [10], [11]. Multimedia servers like *Windows Streaming Media* use the packet pair technique to estimate the bottleneck capacity from the server to the client [12]. Nevertheless, results presented in [13] show that the accuracy of the method is not good when the clients are connected to a WLAN.

Of particular interest to us are the works in [10], [11]. Kapoor *et al* [10] use the CapProbe, Pathrate and PathChar tools to estimate the bottleneck link capacity of a path. The authors of [11] propose the ProbeGap tool to calculate the available bandwidth of the access network in a path. Although one of the reported experiments in [11] includes a WLAN in the path, the result is the effective capacity and not the transmission rate, because the characteristics of the IEEE802.11 protocol are not taken into account. Our work differs from the above in three aspects. First, we take into consideration the characteristics of the IEEE802.11 standard, such as the control overhead and multi-rate adaptation. Second, we estimate a different measure from that of the above work: the actual bit rate of the WLAN device in the last hop. Third, we compute the bit rate of the wireless device even if it is not the bottleneck link in a path.

The rest of the paper is organized as follows. Section II briefly surveys the IEEE802.11 standard and the packet pair technique. Our proposal is presented in Section III and simulation and experimental results are described in Section IV. Section V concludes the paper.

II. Background

A. IEEE802.11 standard

The IEEE802.11 standards [1] define the physical and the MAC layer for a WLAN. The MAC layer defines two access

methods: the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF). The DCF method is based on the CSMA/CA mechanism and provides a best-effort type of service. Since the PCF method is rarely implemented in current 802.11 products, we focus only on the DCF method.

A wireless station can use two mechanisms to access the channel: the basic DCF mechanism and the DCF mechanism with reservation. The reservation scheme adds a significant overhead, then it is typically not used in WLANs with moderate loads or for the transmission of short packets. Therefore we assume the wireless stations use the basic DCF mechanism.

In the basic DCF mechanism, there are specific delays between events: a Distributed Inter-frame Space (DIFS) and a Short Inter-frame Space (SIFS). Suppose that a station wants to send a frame. It senses the channel and, if it is idle, the station transmits its frame after a DIFS. If the destination station receives the frame, it sends an ACK after a SIFS. If another station wants to transmit a message and the channel is not idle, it chooses a random backoff time uniformly distributed between $[0, CW - 1]$, where CW is the actual contention window size in time slots. Usually a time slot is equal to 10 or 20μs. Initially, CW is set to a predefined value CW_{min} which is equal to 32. The station decrements the backoff timer only when the channel is idle, i.e. the timer is frozen if a transmission is detected. If a collision or loss occurs, the contention window of the station is doubled.

The IEEE802.11 standard has support for multi-rate adaptation. The wireless stations can adapt their transmission rates dynamically to achieve a high performance under varying conditions. The standard does not define an algorithm to adapt the transmission rates. However, there are some proposals in the literature that adapt the transmission rate to the channel conditions aiming at optimizing the application throughput or the local power consumption.

B. Packet pair method

The packet pair method is a well-known technique to measure the capacity of a path. It consists of two packets of the same size sent from the source to the destination, one immediately after the other. The method is based on the assumption that the interval between the two packets will be expanded by the bottleneck link and it will be preserved until they arrive at the destination. Suppose that the packet inter-arrival time at the destination host is equal to T and the size of each packet is equal to B. Then, the bottleneck capacity is equal to $C = B/T$.

Some recent work [8], [10], [9] extends the packet pair technique to increase the accuracy of estimating the bottleneck link capacity. They show that the method can produce erroneous results when the cross traffic in the path is high. One solution to improve the accuracy is to send a packet train instead of a packet pair to obtain a histogram of the capacities [8]. Then, the bottleneck capacity is equal to the capacity that has the highest probability. Another solution proposed in [10], [9] is to select the packet pairs with the smallest end-to-end delays.

III. The proposed end-to-end technique

Two main issues have to be addressed for estimating the bit rate of the last hop of a WLAN in a path: the overhead of the 802.11 protocol and the fact that the WLAN may not be the bottleneck.

In order to exemplify the overhead issue, Figure 1 shows two packets (a packet pair) being transmitted in a 802.11 WLAN. In this example we assume the ideal scenario where no cross traffic exists during the transmission of the packet pair in the WLAN. The packet inter-arrival time at the receiver is equal to the sum of the following times: SIFS, DIFS, ACK, backoff and the transmission time of the second packet in the pair. Clearly, the link transmission rate can not be estimated from equation $C = B/T$ (recall that B is the packet size and T is the packet inter-arrival time) used by the original packet pair technique. Since that SIFS, DIFS and ACK transmission times are constant, the equation above can be easily adapted to consider these quantities. However, the backoff time is a random variable that depends on several factors such as the network load. Then it is not trivial to accurately estimate its value.

Note that, even in the absence of concurrent traffic, the backoff time between the transmission of two consecutive messages from the same source can have high variability (see Figure 1) and it is not possible to guarantee that the second packet of the pair will be transmitted within a short interval from the first. This is mainly due to one characteristic of the IEEE802.11 standard: even if the medium is idle the station waits for a backoff time before the transmission of the n-th consecutive message, and this time is uniformly chosen between $[0, CW_{\min} - 1]$. If there is cross traffic, the variability of the interval between two consecutive transmissions can be even higher because the value of the contention window is doubled each time a collision occurs and the backoff time is only decremented when the channel is idle.

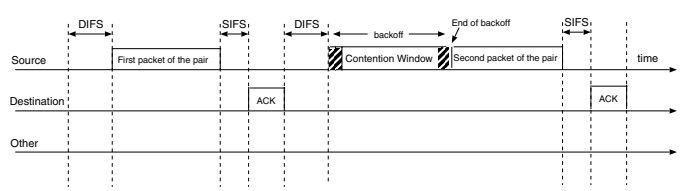


Fig. 1. Transmission of a packet pair in a WLAN using the DCF method.

With respect to the bottleneck issue, because of the high transmission rates of the 802.11a/g standards, it is likely that the wireless link is not the path bottleneck. If one is interested in the bottleneck link capacity of a path, there is no problem if the WLAN is not the path bottleneck. However, for the measure we are interested in, the fact that the last hop WLAN may not be the bottleneck has to be handled.

In summary, because of the above issues, the original packet pair method is not appropriate to estimate the last hop WLAN bit rate. As a consequence, tools like CapProbe [10] and Pathrate [8], both based on the original packet pair technique, can not be employed to our problem. In what follows we present the details of our approach.

Suppose we are interested in measuring the bit rate of a WLAN that connects computer B to the Internet, from computer A connected elsewhere. In order to estimate this measure, A will send probes to B. Consider a sequence of m groups of probe pairs sent from A to B. Each group is formed by four pairs, as shown in Figure 2. Let $\psi_{i,j}^k$ denote a particular packet in the sequence. In our notation, index k ($k = 1, ..., m$) identifies the group (k-th group out of m), j ($j = 1, 2, 3, 4$) is the index of the particular pair in the group and i ($i = 1, 2$) indicates the first ($i = 1$) or second ($i = 2$) packet in any pair. Let $L_{i,j}^k$ be the size of packet $\psi_{i,j}^k$.

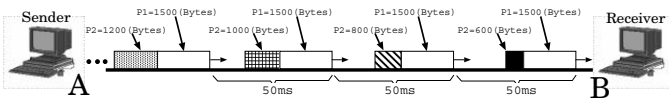


Fig. 2. Packet pairs.

We choose $L_{1,j}^k$ (for any j and k), that is, the size of the first packet in any pair, equal to the maximum transmission unit (MTU) of an Ethernet LAN (1500 bytes) to avoid IP datagram fragmentation. The second packet in a pair has the following sizes: $L_{2,1}^k = 600$, $L_{2,2}^k = 800$, $L_{2,3}^k = 1000$, $L_{2,4}^k = 1200$ bytes, $k = 1, ..., m$. That is, their lengths increase with j, but are all smaller than the first packet in a pair.

The reasoning about the choices for $L_{i,j}^k$ is as follows. Assume initially that there is no concurrent traffic in the path from A to B and the packet pair follows the same path until it reaches the receiver. In this case, it is easy to see that, at any hop from A to B, except the last (WLAN), the transmission of the second packet will always starts immediately after the first packet is sent, because of the large size of the first packet as compared to the first.

Consider a path with n hops, each with capacity C_l, $l = 1, ..., n$. Let Δ_l be the dispersion of a packet pair at the l-th hop assuming both packets in the pair have **the same size** L. Δ_l is given by [8]:

$$\Delta_l = \max_l \{\Delta_{l-1}, L/C_l\}. \quad (1)$$

Then, the dispersion at the receiver can be computed from $\Delta_{rec} = L/C_{min}$, where C_{min} is the capacity of the narrowest link in the path [8].

In our proposal, probe packets have different sizes, and the second packet in a pair has a smaller size than the first. The rationality behind the proposal is that the first packet "slows down" the second increasing the likelihood that they will arrive back to back at the receiver.

Suppose A sends a pair of packets to B. As the pair traverses the links from A to B, it is likely that it encounters links with increasing speeds as the packets move towards the network core. As the pair leaves the network core and moves towards the edge, the link capacities are likely to be reduced and the pair may join again near the last hop.

To clarify the idea, consider, for example, a scenario where $L_{1,j}^k/C_{l+1} \geq L_{2,j}^k/C_l$, that is, the second packet arrives at hop $l+1$ before or immediately after the first packet is transmitted from that hop. Then the dispersion $\delta_{l+1,j}^k$ at the (l+1)-th hop of the j-th packet pair in the k-th group is:

$$\delta_{l+1,j}^k = L_{2,j}^k/C_{l+1}, \text{ for } j = 1, 2, 3, 4; \; k = 1, ..., m. \quad (2)$$

Clearly, for this scenario, the dispersion of the j-th pair at hop $l+1$ is the transmission time of the second packet at that hop. Then, in the last hop, the dispersion at the receiver is equal to the transmission time of the second packet.

As an example, consider a scenario with cross traffic where the path from the source to the destination has four hops and the last hop is an IEEE802.11g WLAN. This scenario is identical to the first experiment described in section IV and it is used here with the purpose of emphasizing the main attributes of the algorithm.

Figure 3 shows the results of an experiment where 40 packet pairs were generated using our algorithm. In the figure we plot the dispersion computed at the receiver for each of the 40 pairs ordered by index j, i.e. the points in the same vertical line represent pairs in which the second packets have identical sizes. (The x-axis in the figure shows the size of the second packet in a pair, instead of index j.) Note the large variability of the dispersion values for packets with the same index j. This variability is mainly due to the concurrent traffic in the path and the backoff time of the IEEE802.11 standard.

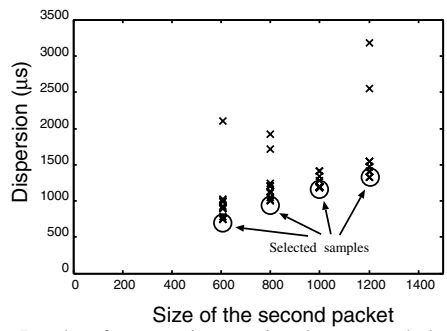


Fig. 3. Results of an experiment using the proposed algorithm.

Our goal is to reduce the effects of the backoff time and cross traffic. Then, for each j, we select the smallest dispersion value in the samples. If m groups are generated from the source to the destination, we select δ_j as:

$$\delta_j = min_{k=1,m}\{\delta_{N,j}^k\}, \; j = 1, 2, 3, 4 \quad (3)$$

where N is the index of the last hop in the path (the WLAN).

In a idealized scenario without cross traffic and backoff times, the dispersion values δ_j ($j = 1, \ldots, 4$) are a function of the following times: SIFS and DIFS (which are constants), transmission of the ACK and of the second packet in a pair. As a consequence, the dispersion is a linear function of the bit rate of the WLAN link.

Let t_{SIFS} and t_{DIFS} be the length of SIFS and DIFS intervals, respectively. The ACK transmission time is L_{ACK}/C_n and L_{2j}/C_n is the transmission time of a packet with length equal to that of the second packet in the j-th pair in a group. Assuming that packets arrive back to back in the WLAN device, the dispersion is $D_{j,C_n} = t_{SIFS} + t_{DIFS} + L_{ACK}/C_n + L_{2j}/C_n$.

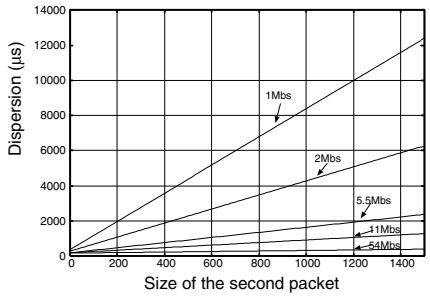


Fig. 4. Lower bound for the dispersion in a IEEE802.11 standard.

Figure 4 plots the dispersion D_{j,C_n} for each 802.11 capacity and L_{2j} values. These plots will be used as the basis for comparisons against the measured data. That is, the final step of our algorithm consists of calculating the Mean Squared Error (MSE) between the measured δ_j and D_{j,C_n} for each value of C_n. The measure we want to estimate, C_{WLAN} is obtained from:

$$C_{WLAN} = \min_{\forall C_n}\{MSE(\delta_j, D_{j,C_n})\} \quad (4)$$

In summary, our technique has five-steps described bellow:

Algorithm 1

Step 1: Using any technique (e.g., [2], [5]), identify the connection type of the last hop. If it is a WLAN, then continue Steps 2-5;
Step 2: Generate a sequence of m groups of packet pairs and collect it at the receiver;
Step 3: At the receiver compute the dispersion $\delta^k_{n,j}$ for all $(4 * m)$ pairs, where $k = 1, ..., m$ is the group index and $j = 1, 2, 3, 4$ is the index of a particular pair in the group;
Step 4: Using equation 3, select the smallest dispersion value for each value of $j = 1, 2, 3, 4$ and obtain δ_j;
Step 5: Estimate the C_{WLAN} using equation 4.

As mentioned in section II, it is possible that a wireless station changes its transmission rate according to the wireless link conditions. The method we propose can also be employed for dynamically estimating the changes. We generate continuously groups of four packet pairs during the estimation period of interest. For the transmission rate estimation we use a window of W packet pairs and apply the algorithm above (in this case $m = W/4$). For the next estimation, the window slides by β packet pairs. The new β pairs replace old pairs and the algorithm is re-applied. Figure 5 shows an example of the dynamic computation procedure at the receiver. At each instant t_i, a new value C^i_{WLAN}, $i = 1, 2, ...$, is obtained.

There is a clear trade-off between the value of W and accuracy. The window size W (i.e. the number of probe pairs in an estimation interval $(t_i, t_i - W)$) has to be *large enough* to obtain accurate results. If the interval $(t_i, t_i - W)$ is small, then the probe generation rate must increase so that a sufficient number of probes are collected in an estimation interval. The parameter β determines the frequency at which transmission rates are computed. If it is equal to one, a new rate value is computed whenever a new packet pair arrives. The smallest the value of β the fastest the changes in the transmission rate are captured. In section IV we further discuss these issues.

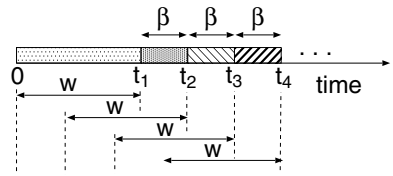


Fig. 5. Dynamic computation of the transmission rates.

IV. Validation

We evaluate the accuracy of the proposed technique using both experimentation and a NS-2 [14] simulation model. For the experiments we used an IEEE802.11 access point transmitting at a constant data rate. In the simulations the access point was configured to operate using multi-rate adaptation.

A. Validation via experimentation

In the first set of experiments, the access point was configured to operate at a low, moderate and high transmission rates. The wireless connection was the bottleneck link in the path between the probe transmitter and the receiver. In the second set of experiments, the bottleneck *was not* the WLAN. In each experiment the transmitter sent 40 packet pairs every second during 10s. (Since k identifies a group of 4 packet pairs, k varies from 1 to 100.)

The first set of experiments was performed in our University (UFRJ) Lab. Figure 6 shows the topology used. It consists of two sources (A1 and A2) connected to a 100Mps Ethernet switch port and two receivers (B1 and B2) connected to an IEEE802.11g WLAN. All the machines run LINUX. Traffic from the sources to the receivers traverses two routers: the department router and our Lab router. Packet probes were generated from A2 to B2, and simultaneously, three FTP connections were established from B1 to A1 to produce concurrent traffic. The wireless access point was configured to operate at different transmission rates. All other links have capacity equal to 100Mbps.

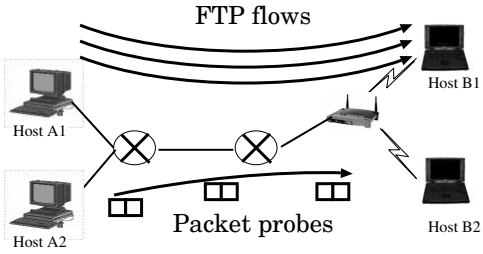
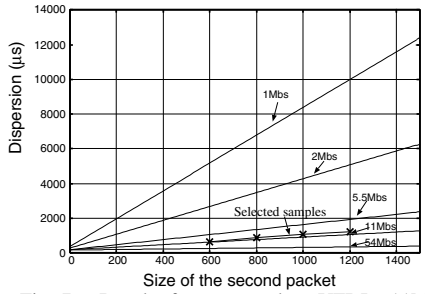


Fig. 6. Network used in the first set of experiments.

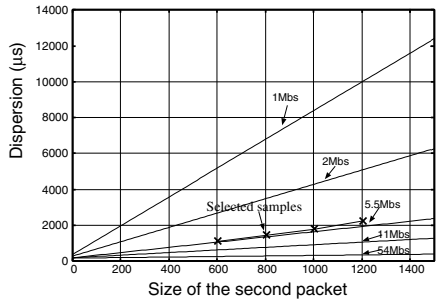
Figure 7 shows the dispersion values computed from our algorithm when the access point was configured to operate at 11Mbps. The figure also shows the MSE values calculated in accordance with the last step of the algorithm. Clearly, the minimum MSE value is obtained for the 11 Mbps transmission rate. This rate can also be inferred if we visually compare the line connecting the dispersion samples with the curves plotted in Figure 4.



Rate	MSE
1	48194925
2	10231681
5.5	824976
6	639383
9	286762
11	=>244734<=
12	494648
18	283931
24	337879
36	413688
48	459787
54	476372

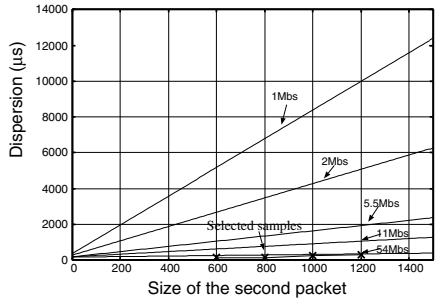
Fig. 7. Results for a network at UFRJ - 11Mbps wireless link.

We considered others values for the access point transmission rate. For all values, the algorithm correctly estimated the WLAN bit rate. Figure 8 and 9 show the results where the access point was configured to operate at 5.5Mbps and 54 Mbps. From these experiments we conclude that the accuracy of our algorithm is not affected by the transmission rate of the wireless device.



Rate	MSE
1	33682846
2	4161902
5.5	=>127568<=
6	242075
9	827492
11	1082290

Fig. 8. Results for a network at UFRJ - 5.5Mbps wireless link.



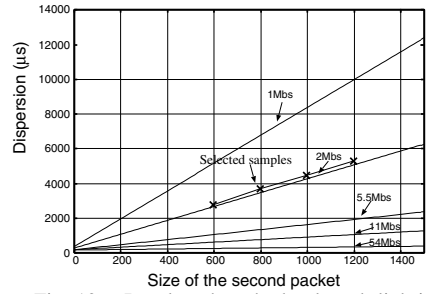
Rate	MSE
12	360024
18	154570
24	84633
36	36558
48	20719
54	=>16654<=

Fig. 9. Results for a network at UFRJ - 54Mbps wireless link.

In the second scenario the source-destination path goes through a home access link, which is a 512Kbps cable modem, and the receiver is connected to that link via a wireless channel. In this scenario the bottleneck is the home access link and not the WLAN. The sender machine was in our Lab while the receiver is in a home in Rio de Janeiro. The path had eleven hops, and the access point was configured to operate at 2Mbps. Figure 10 plots the four dispersion values calculated from our algorithm. From the MSE values presented in the figure, the algorithm correctly estimated the 2Mbps bit rate.

B. Validation through simulation

The topology of the NS-2 simulation model is shown in Figure 11. Nodes *S1* and *S2* represent the traffic sources and *W1* and *W2* are the destinations. The paths from *S1* and *S2* to *W1* and *W2* have three wired links and the last hop is an 802.11. The capacities of links *L1*, *L2* and *L3* are equal



Rate	MSE
1	10980400
2	=>208884<=
5.5	8106042
6	8895886
9	11564614
11	12469772
12	13030199
18	14582867
24	15392299
36	16223556
48	16647337
54	16789594

Fig. 10. Results when the bottleneck link is not the WLAN.

to 100Mbps. The link *L4* between router *R2* and the access point has capacity equal to 10Mbps. The *L4* transmission rate was chosen 10Mbps to evaluate scenarios where the WLAN was not the path bottleneck. The propagation delay is equal to 10ms for all links. Hosts *S2* and *W2* are the source and receiver, respectively, for the probes. Source *S2* generates the probes while three FTP connections were simultaneously established from *W1* to *S1* to simulate cross traffic.

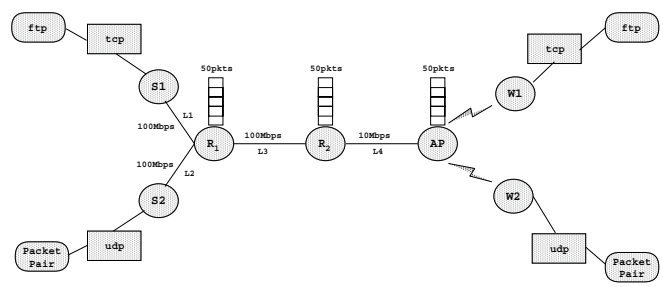


Fig. 11. Simulation model.

In order to simulate the multi-rate adaptation of the IEEE802.11 standard, we carried out three experiments using an IEEE802.11g access point in our laboratory, configured to operate with multi-rate adaptation. To collect data for each experiment, a student with a laptop connected to the access point freely walked around our laboratory and sampled the device bit rate. For the first and second experiments, the samples were taken at every second during 5 minutes For the last experiment, they were taken every 30 sec during 25 minutes. The collected data were then used in the simulation model to mimic the wireless device dynamic rate adaptation.

The value of the parameters used for the proposed algorithm are: (i) the sender generates probes at a rate 20 packet pairs per second; (ii) $W = 20$, $\beta = 1$ in the first and second experiments; (iii) $W = 160$, $\beta = 1$ in the third experiment. Using the above parameter values, a new estimation for C_{WLAN} is computed at every $1/20$ sec.

Figures 12(A) and 13(A) shows the results for the first and second simulations, respectively. From the figures we observe that the transmission rate estimated by the algorithm (solid line) and the real bit rates collected during the experiment (dashed line) are very close. Note that the algorithm is able to accurately capture the dynamic behavior of the device bit rate. Only for a small number of intervals the estimated bit rate differs from the real rate, for instance, around $t = 12s$. In this case, the estimated rate is 36Mbps while the real rate is 54Mbps. This error may be attributed to cross traffic and the

IEEE802.11 backoff time, since they increase the packet pair dispersion lowering the estimated rate.

The relative error is depicted in Figures 12(B) and 13(B). From the figures we see that relative errors of more than 70% occur less than 10% and only 20% of the estimations have a relative error greater than 20%.

Figure 14(A) shows the estimated and real rate and Figure 14(B) the relative error for the third simulation. In this case the algorithm uses more samples to compute the transmission rate. In the previous models each estimation is based on a 20 packet pairs window while in the third experiment $W = 160$. We can see from the figures that the algorithm is more accurate in this scenario than in the previous two. 87% of the estimations have a relative error less than 20%.

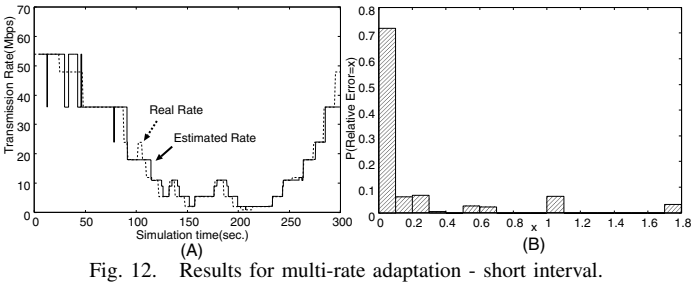


Fig. 12. Results for multi-rate adaptation - short interval.

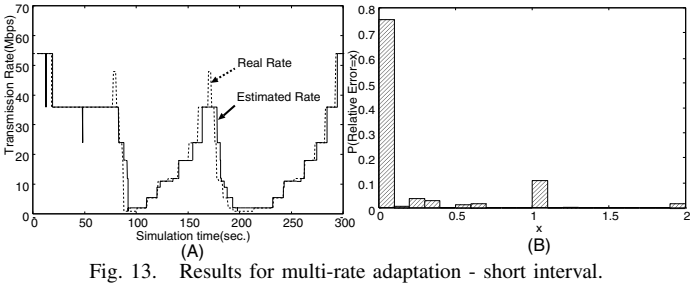


Fig. 13. Results for multi-rate adaptation - short interval.

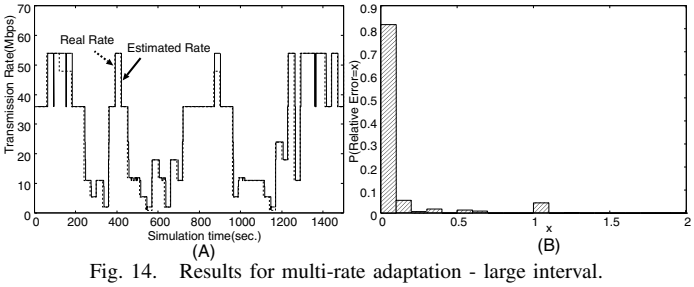


Fig. 14. Results for multi-rate adaptation - large interval.

Finally, we note that in the three considered scenarios, the algorithm is more accurate to detect a transition from a low to a high transmission rate than otherwise. This is explained as follows. Suppose that the transmission rate increases from 5.5Mbps to 11Mbps. Consider also that, at the moment the algorithm computes the new transmission rate, there are still samples of dispersion that were collected when the rate was 5.5Mbps. Although these *old* dispersion values are considered, they will not be selected by the algorithm because the are greater than those recently computed. (Note that the new dispersion values were obtained from a packet pair transmitted using a 11Mbps data rate.) On the other hand, if the real transmission rate decreases, the *old* samples are likely to be selected because they have smaller values than those for the recent samples. As a consequence, transmission rate will be overestimated during a short time interval.

V. Conclusions

The development of end-to-end techniques to infer the transmission rate of the access network when it is a WLAN can be very useful for several applications. In this work, a simple and accurate active measurement technique to infer this metric is proposed. The technique is an extension of the packet pair method and takes into account the overhead caused by IEEE 802.11 and the existence of cross traffic. Results from measurements and simulation show that the accuracy of the method is very good both in scenarios where the WLAN devices uses fixed transmission rates and in scenarios in which the devices have support to any auto rate control mechanism.

Acknowledgment

This work is supported in part by grants from CAPES, CNPq and FAPERJ.

References

[1] IEEE standard 802.11a/b/g, "IEEE 802.11, 802.11a, 802.11b, 802.11g standards for wireless local area netowrks," http://standards.ieee.org/getieee802/802.11.html.

[2] W. Wei, B. Wang, C. Zhang, J. Kurose, and D. Towsley, "Classification of Access Network Types: Ethernet, Wireless LAN, ADSL, Cable Modem or Dialup?" in *IEEE Infocom*, 2005.

[3] M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L.Wang, "TCP over wireless multihop protocols: Simulation and experiments," in *IEEE ICC*, 1999.

[4] R. Cohen and S. Ramanathan, "TCP for high performance in hybrid fiber coaxial broad-band access networks," in *IEEE/ACM Transactions on Networking*, vol. 6, 1998, pp. 15–29.

[5] L. Cheng and I. Marsic, "Fuzzy Reasoning for Wireless Awareness," in *International Journal of Wireless Information Networks*, vol. 8, 2001, pp. 15–26.

[6] V. Jacobson, "Congestion Avoidance and Control," in *ACM Sigcomm*, 1988, pp. 314–329.

[7] S. Keshav, "A Control-Theoretic Approach to Flow Control," in *ACM Sigcomm*, 1991, pp. 3–15.

[8] C. Dovrolis, P. Ramanathan, and D. Moore, "What do Packet Dispersion Techniques Measure?" in *IEEE Infocom*, vol. 1, 2001, pp. 905–914.

[9] A. Rocha, R. Leao, and E. de Souza e Silva, "A New Technique to Select Packet Pairs to Estimate Bottleneck Link Capacity (in portuguese)," in *III WPerformance/XXIV SBC*, 2004.

[10] R. Kapoor, L. Chen, L. Lao, M. Gerla, and M. Sanadidi, "CapProbe: A Simple and Accurate Capacity Estimation Technique," in *ACM Sigcomm*, vol. 34, 2004, pp. 67–78.

[11] K. Lakshminarayanan, V. Padmanabhan, and J. Padhye, "Bandwidth Estimation in Broadband Access Networks," in *ACM SIGCOMM Internet Measurement Conference(IMC)*, 2004, pp. 314–321.

[12] J. Nichols, M. Claypool, R. Kinicki, and M. Li, "Measurements of the Congestion Responsiveness of Windows Streaming Media," in *NOSSDAV*, 2004, pp. 189–202.

[13] F. Li, J. Chung, M. Li, H. Wu, M. Claypool, and R. Kinicki, "Application, Network and Link Layer Measurements of Streaming Video over a Wireless Campus Network," in *Passive and Active Network Measurement Workshop (PAM)*, 2005, pp. 189–202.

[14] NS-2, "The Network Simulator ns-2. http://www.isi.edu/nsnam/ns/."