Practical Rate Adaptation for IEEE 802.11 WLANs

Qiuyan Xia, Mounir Hamdi and Tsz Ho Chan
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Kowloon, Hong Kong, China
Email: {xiaqy, hamdi, cscth}@cse.ust.hk

Abstract— **Wireless Local Area Networks (WLANs) have become increasingly popular due to the recent availability of affordable devices providing multirate capabilities. However, because of the time-varying characteristics of a wireless channel, no single rate can be optimal under all scenarios, and the device needs to tune its transmission rate dynamically. Therefore, rate adaptation is a critical component of its performance. In addition, the IEEE 802.11 standard does not specify an algorithm for automatic rate selection; it is intentionally left open to the vendors. In this paper, we propose a Practical Rate Adaptation algorithm, which utilizes both statistics and RSSI of ACK to decide the transmission rate that maximizes the throughput. We implement our algorithm in commercial products and carry out extensive experiments for performance evaluation. The results demonstrate that using both statistics and RSSI of ACK greatly improves system throughput and responsiveness under various wireless environments.**

I. Introduction

In recent years, WLAN technology has evolved at a rapid speed. Most of the commercial WLAN products are based on the IEEE 802.11 standard [1]. The physical layers of 802.11 provide multiple data rates; for example, 802.11a [2] defines 8 different data rates ranging from 6 up to 54 Mbps. Higher data rates are achieved by more efficient modulation schemes. A high level modulation can be used when the channel Signal-to-Noise Ratio (SNR) is sufficiently high such that the received signal can be properly decoded. Therefore, a tradeoff emerges between the data rate and the Bit Error Rate (BER).

With the multirate capability, it is desirable to always transmit data at the highest possible rate given current channel conditions. However, in wireless systems, the radio propagation environments vary over time and space due to such factors as signal attenuation and fading, motion of objects, interference and so on, causing variations in the received SNR. As a result, there is no single modulation that can be optimal under all scenarios. As the multirate enhancements are the PHY layer protocols, the MAC layer mechanism is required to exploit this capability. The problem of dynamically selecting an appropriate transmission rate out of multiple available data rates is referred to as *Rate Adaptation*. In IEEE 802.11, rate adaptation algorithm is intentionally left open. The standard does not specify when and how to switch among different data rates. In addition, no signaling mechanism is available in the standard for the receiver to notify the sender about the channel conditions. To determine the best transmission rate at a given time, the sender needs to know the Channel State Information (CSI) in advance ideally, i.e., SNR, and Frame Error Rate (FER) vs. SNR relationship for different transmission rates, both at the receiver side. In reality, neither of them is known prior to the sender, and both are time-varying factors. The sender has to decide the transmission rate based on limited CSI feedback such as ACK, retry count and FER.

Our objective in this paper is to design a Practical Rate Adaptation (PRA) algorithm at the MAC layer, which works well in a variety of channel conditions; besides, no protocol modification is required so that it can be easily deployed with current wireless devices. Specifically, PRA dynamically monitors and adapts to changes in the link quality by tuning the packet transmission rate, in order to maximize the throughput. Moreover, to improve system responsiveness, PRA uses the Receive Signal Strength Indicator of ACK (RSSIA) to predict the dynamics of the receiver-perceived link conditions for rate adaptation at the sender. We show that using these methods can significantly improve system performance.

The rest of the paper is organized as follows. In section II, we briefly introduce the IEEE 802.11 standard and related work. The basic ideas and practical implementation of the proposed algorithm are presented in section III in detail. Section IV gives extensive experiment evaluation of our algorithm compared with other state-of-the-art algorithm. Finally, this paper concludes with section V.

II. Background and Related Work

A. IEEE 802.11 WLAN

The IEEE 802.11 standard specifies the Medium Access Control (MAC) and Physical (PHY) layers for a WLAN system. Two medium access mechanisms are defined in 802.11: the Distributed Coordination Function (DCF) is a mandatory contention-based access protocol; the Point Coordination Function (PCF) is a priority-based contention-free protocol. There are currently three PHY layer extensions: 802.11b, 802.11a, and 802.11g. Since the PCF is rarely used in current WLAN devices, in this paper, we focus on rate adaptation for 802.11a WLANs based on the DCF. The 802.11a PHY provides eight PHY modes with different modulation schemes and coding rates: 6, 9, 12, 18, 24, 36, 48 and 54 Mbps.

B. Rate Adaptation Algorithms

Several rate adaptation algorithms for 802.11 WLANs have been proposed in the literature. Based on the CSI used for channel quality estimation, they can be roughly divided into two categories: *statistic-based* and *signal-measurement-based*

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

schemes. TABLE I gives the comparisons between them. Readers can refer to these papers as well as [3] [4].

III. Practical Rate Adaptation Algorithm

Those receiver-based rate adaptation algorithms, which assume some communication of rate selection between the sender and the receiver like RBAR [10], usually result in better system performance than the sender-based ones. However, the common weakness is that they are incompatible with the current 802.11 standard. In this paper, we focus on designing practical sender-based algorithms with an "ignorant" receiver, that can still be self-tuning and fast responsive. Our scheme is based on the following observations:

- The static nature of some previously proposed algorithms (using static thresholds) makes them less versatile to different channel conditions;
- The data rate that can provide the highest throughput should be used, no matter how lossy it may be [9];
- Signal strength can be helpful in statistic-based algorithms, but should not be directly used for rate selection.

Specifically, the proposed PRA algorithm collects statistics and predicts channel dynamics based on the limited CSI feedback. The key of this scheme is when to change the rate, and which rate to switch to. To achieve our goals, both statistics and signal strength are incorporated in the rate adaptation. The former are continuously collected and updated; based on them, an optimization metric, i.e., the throughput, is calculated and acts as a primary rate-switch-decision trigger. The latter aids to safeguard the selected rate, and improve system responsiveness.

A. Introduction to the AR5212 Chipset and MADWIFI

We develop and implement our algorithm in the Linux driver MADWIFI [8], for wireless adapters based on the Atheros AR5212 chipset [14]. The chipset only implements time-critical functions; less critical ones, including rate adaptation, are shared between the MAC controller and the host system. It maintains several FIFO (First In First Out) queues of transmission descriptors to schedule packets for transmission. Each descriptor contains control information for a frame's transmission and a status field that records how the transmission is completed. Among them, the most relevant ones are the "multirate retry series", and the status field including the subfields of "ok", "excessive retries", "fail count", "final transmit index", "ACK signal strength" and so on. There are four retry series $(r_0/c_0, r_1/c_1, r_2/c_2, r_3/c_3)$, each pair meaning that attempts at rate r_i is at most c_i times. Users can either disable this function, or customize the retry series. Whenever data is received from the upper layer, the driver prepares a descriptor for it by filling in the fields with proper initial values, and inserts the descriptor into one of the FIFO queues. As soon as the medium is available, the HOL (Head-of-Line) frame is sent at the rate r_0; other retries are automatically carried out if necessary, as specified by the multirate retry series. Finally, the transmission status is returned to the descriptor for the reference of users of the driver.

The driver already implements an ARF [5]-like rate control module, namely, the ONOE algorithm. It involves two phases of adjustment. The short-term variations are handled by the multirate retry mechanism, i.e., it sets up two step-down retry rates and make the lowest rate the "last chance": $r_1 = r_0 - 1$, $r_2 = r_1 - 1$ and $r_3 = 0$. Here c_0, c_1, c_2 and c_3 are set to 4, 2, 2, 2 respectively. The long-term variations are handled by changing the values of multirate retry series at regular fixed intervals. To do so, ONOE runs periodically (1000 ms) analyzing transmit statistics for each destination. If transmissions at the current rate are judged to be good in that interval, a "tx_upper credit" is issued; otherwise, the credit counter is deducted. If the total credits at the current rate exceeds the $rate\_raise\_threshold$ (10), the transmission rate is raised. If several error conditions occur, the transmission rate is decreased to the next lower one. Whenever the rate is changed, the credit counter and the statistics are reset. We note that though ONOE is much less sensitive to individual packet failure than ARF, it is also more conservative. When a better channel occurs, it takes at least 10 seconds to scale up. Similarly, if the channel is deteriorating, it can only step down to the next lower rate for each interval and eventually, there are too many packet losses before it finds a proper rate.

Recently, another rate control algorithm, SampleRate [9], has been implemented. SampleRate periodically sends packets at rates other than the current one to estimate whether another rate will offer better throughput. Then it chooses the rate which may provide the most throughput based on estimates of the expected per-packet transmission time at each rate.

B. Implementation of PRA in MADWIFI

1) Multirate retry mechanism: PRA is still a two-phase process. The multirate retry deals with transient channel variations, with modified c_i values for our rate adaptation goals. Among the four series, the allowed attempts for the first retry series c_0, is most relevant to system responsiveness. A high value makes the second and later attempts use the same rate as the first, which may in turn cause additional retries in the case of a deteriorating channel. Therefore, we set c_0 to 2, which means that the first and second attempts use the long-term rate r_0; after that, the possible following retries use lower rates. Similarly, to ensure fast responsiveness to short-term channel variations, c_1, c_2, c_3 are all set to a smaller value of 1.

2) State transitions: The long-term transmission rate for a node, $txRate$, is adjusted according to its current state. Fig. 1 illustrates the state transitions. Specifically, a sender has two states, "Tx" and "Probe". The statistics for these states are updated regularly according to a timer. At the beginning of a new round, the sender is at its "Tx" state and transmits packets at $r_0 = txRate$. Meanwhile, the sender monitors the transmission results of the current rate and if it observes a promising rate by $may\_probe()$, which may improve the throughput, the sender sets up a new $probeRate$ and enters into the "Probe" state. The following arriving packets are sent using $r_0 = probeRate$. Similarly, the sender keeps on monitoring the probing results at the $probeRate$ and if it

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

TABLE I

COMPARISONS BETWEEN STATISTIC-BASED AND SIGNAL-MEASUREMENT-BASED SCHEMES

Statistic-based schemes	Signal-measurement-based schemes
ARF [5], Dynamic ST [6], AARF/AMRR [7], ONOE [8], SampleRate [9]	RBAR [10], Goodput analysis [11], OAR [12], RSS measurement [13]
collect transmission statistics	measure the signal strength
no requirement for RTS/CTS and no changes to the standard	may demand RTS/CTS and entail changes to the standard
have to be carefully designed to achieve good performance	good performance (if neglect the overhead)

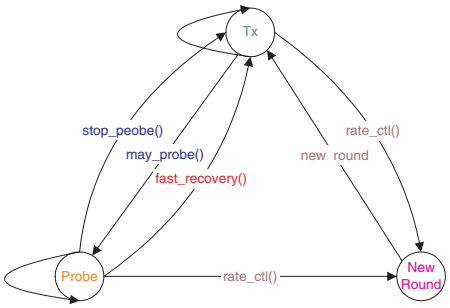


Fig. 1. State transition at the sender.

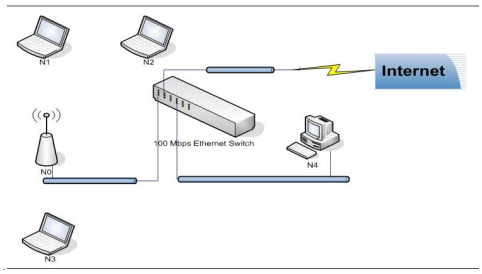


Fig. 2. Network topology in the experiments.

TABLE II

EXPERIMENT PARAMETERS

FT_{Max}	6	ST_{Max}	50
FT_{Min}	4	ST_{Min}	8
α	2	UDP Packet	1472 Bytes
β	6	Simulation time	120 s
$enough$	20 packets	c_0, c_1, c_2, c_3	2, 1, 1, 1

concludes that the $probeRate$ offers a better performance than the $txRate$, the sender quickly recovers by setting $r_0 = txRate = probeRate$ and enter into a new "Tx" state; otherwise, the sender stops probing and goes back to the "Tx" state. After settling r_0, the remaining rate series r_1, r_2, r_3 are determined in the same way as in ONOE. Whenever a state transition occurs, the associated statistics are reset. This process continues until the timer expires, then the statistic counters are reset and a new round is started. The three procedures, $may\_probe()$, $stop\_probe()$ and $fast\_recovery()$ describe when the sender is allowed to change its current state, which is critical to the throughput performance.

Algorithm 1 provides the pseudo code description of the proposed algorithm. In the PRA algorithm, the sender collects statistics in $tx\_complete()$. The average RSSIAs are used to bound a feasible rate, and predict channel dynamics, which are implemented in the functions of $lookup\_rssiThresholdTable()$, $fast\_up()$ and $fast\_down()$ respectively. Note that at the "Tx" state, the sender also uses a threshold-based scheme like in ARF, with FT_{Min} set to 4. However, to resolve the problems of ARF, it uses MILD to dynamically adapt ST, which is bounded in the interval $[ST_{Min}, ST_{Max}]$. Whenever the threshold is reached (along with some other conditions), the sender starts the "Probe" state by setting $curRate = probeRate$. The rate selection is made in the function $findrate()$, which decides the series 0 rate r_0 and retry number c_0 for a new outgoing packet. With the multirate retry mechanism enabled, r_0 is set to the current rate $curRate$, which is either the long-term transmission rate $txRate$, or the probing rate $probeRate$. Then, the other multirate retry series are decided in the function $setupxtxdesc()$, which completes the initialization of the transmission descriptor for that packet. Finally, the HOL packet is transmitted when the medium becomes available.

IV. PERFORMANCE EVALUATION

A. Experiment Settings

We consider the network topology shown in Fig. 2. It consists of three laptops (N_1, N_2, N_3), all equipped with wireless adapters based on the AR5212 chipset, and running the MADWIFI device driver. They are configured to work under 802.11a and communicate with each other through the AP (N_0). The AP is connected to a 100 Mbps Ethernet switch, which extends the wireless LAN to a wired LAN, where a PC (N_4) is located. All equipment are placed in a typical office environment with lots of concrete walls and moving objects. We use the network performance benchmark, "Netperf" [15], to generate continuous saturated UDP traffic and report the achieved application level throughput. TABLE II summarizes the parameters used in the experiments.

B. Experiment Results

In this section, we present the experiment results of the ONOE and the PRA algorithms, subject to different wireless

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

Algorithm 1 PRA Rate Adaptation Algorithm

```
1: tx_complete() :
2: update try counter of packet transmissions at each rate (try[ ]);
3: update average RSSIs of recent ACKs (rssi);
4: bestRate = find_best_rate(try[ ]);
5: feasibleRate = lookup_rssiThresholdTable(rssi_0);
6: if status == 0 then
7:    err = 0;
8:    if retryCount == 0 then
9:       success++;   failure = 0;
10:   else
11:      success = 0;   failure++;
12:   end if
13: else
14:   success = 0;   failure = 0;   err++;
15: end if
16: if !probe && prRate = may_probe()! = −1 then
17:   curRate = probeRate = prRate;   probe = 1;
18: else if probe && fast_recovery() then
19:   curRate = txRate = probeRate;   probe = 0;
20: else if probe && stop_probe() then
21:   curRate = txRate;   probe = 0;
22: end if
1: findrate() :
2: if mrretry  then  rix = curRate;   try0 = 2;
3: else  rix = fixedRate;   try0 = TXMAXTRY;
4: endif
1: setupxtxdesc() :
2: rate0 = curRate;   rate3 = 0;
3: rate1 = −−rate0 > 0?rate0 : 0;
4: rate2 = −−rate0 > 0?rate0 : 0;
1: may_probe() :
2: prRate = −1;
3: if !maxRate() && success ≥ ST_Min then
4:    if feasibleRate > txRate then
5:       prRate = feasibleRate;
6:    else if fast_up(rssi) || success ≥ ST then
7:       prRate = txRate + 1;
8:    end if
9:    if prRate > txRate then
10:      recovery = 1; {mark the first probe}
11:      ST* = α;   ST = min(ST, ST_Max);
12:   else
13:      recovery = 0;
14:   end if
15: end if
16: if success == 0 then
17:   if !minRate()&&failure ≥ FT_Min then
18:      if feasibleRate < txRate &&
         (fast_down(rssi) || failure ≥ FT_Max ||
         avgRetry ≥ 2) then
19:         prRate = txRate − 1;
20:      end if
21:   else if err > 0 then
22:      prRate = bestRate;
23:   end if
24:   if recovery then
25:      ST* = α;   ST = min(ST, ST_Max);
26:   else if txRate > prRate then
27:      ST− = β;   ST = max(ST, ST_Min);
28:   end if
29:   recovery = 0;
30: end if
31: if prRate! = −1 && canProbe[prRate] == −1 then
32:   prRate = −1;
33: end if
34: return prRate;
```

TABLE III
EXPERIMENT SCENARIOS

Channel quality	good	poor	good	poor
N_0 stationary while N_1	stationary	stationary	moving	moving
UDP traffic ($N_1 \rightarrow N_0$)	1	2	3	4

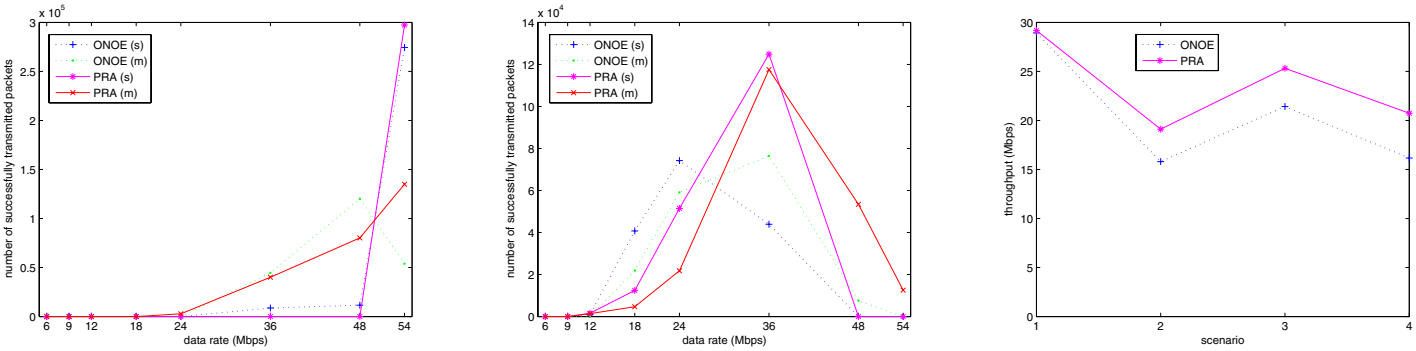
environments shown in TABLE III. The index is used to represent a particular scenario.

Scenario 1 to 4 study the UDP performance of ONOE and PRA (Fig. 3). In all scenarios, PRA manages to send more packets at higher rates than ONOE, mainly due to the "probe" and "fast recovery" mechanisms. In scenario 1, both ONOE and PRA can achieve similar optimal throughput (Fig. 3(a)). However, ONOE spends a much longer time, 10 seconds at the initial rate 36 Mbps before scaling up to 48 Mbps and spends another 10 seconds at 48 Mbps before finally reaching the best 54 Mbps; on the other hand, PRA can quickly recover to 54 Mbps after several packet transmissions at the beginning of the first second. Therefore, PRA still achieves better performance than ONOE. In scenario 2 (Fig. 3(b)), the channel quality is poor and the optimal long-term rate is around 24 and 36 Mbps. However, since ONOE requires 10 credits to trigger a rate increase, it spends two much transmission time at 24 Mbps and can only transmit at 36 Mbps for less time. PRA does not restrict the rate increase action and can respond to the variations quickly enough (several packet-transmission time). So the packets sent at 36 Mbps is much more than in ONOE. Similarly, in scenario 3 (Fig. 3(a)), the channel quality is good but N_0 is moving around, causing the received signal strength to vary over time at the AP. Therefore, ONOE can hardly accumulate 10 credits to scale up at 48 Mbps. However, when PRA observes an improving channel, it quickly probes at 54 Mbps, and when it senses that the channel quality is deteriorating, it either stops probing or retracts to 48 Mbps immediately. This is also true for scenario 4. The achieved UDP throughput is given in Fig. 3(c).

Fig. 4 shows the trace profiles of SampleRate and PRA. For SampleRate, the rate decreases when the channel quality drops for certain interval, but with a phase lag. Also, the rate decreases are not smooth but with lots of glitches. However, with PRA, the sender can predict and tune to the channel variations quickly with little delay. PRA is opportunistic, which can quickly switch to a higher rate once a good channel occurs, and down-scale while observing a string of failures.

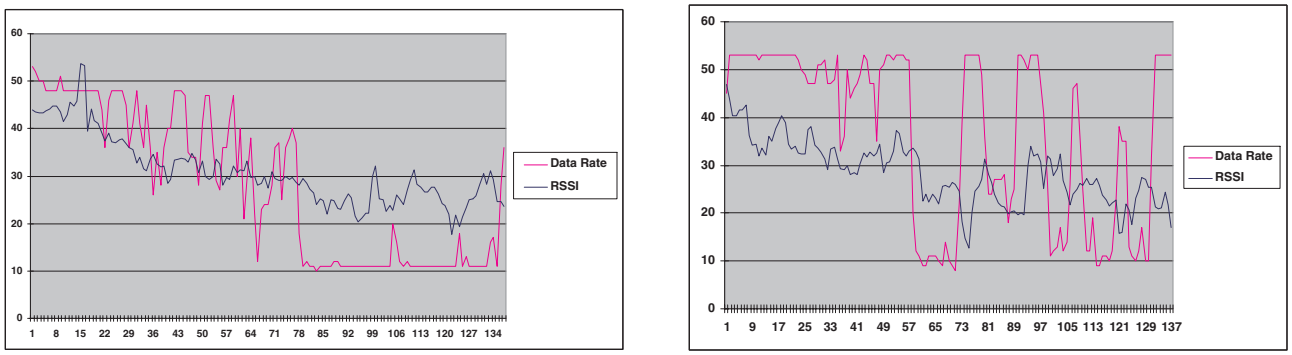
Another feature of PRA is, though it reacts to the result of single packet transmission, it is still safe enough, by using RSSIA to bound the rate switches. Moreover, in PRA, the dominant factor to decide a rate switch is the throughput metric. If the current long-term rate can offer better throughput, no adjustment for the long-term rate is made though it may be suboptimal under some transient channel conditions; indeed, the short-term variations are resolved by the "multirate retry". In the case where PRA may select a wrong rate to probe, it can perceive the error through a few packet transmissions and

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.



(a) Scenario 1 and 3: good channel quality (b) Scenario 2 and 4: poor channel quality (c) UDP throughput in Scenario 1-4

Fig. 3. UDP $N_1 \rightarrow N_0$: number of successfully transmitted packets at each data rate.



(a) SampleRate trace (b) PRA trace

Fig. 4. UDP $N_1 \rightarrow N_0$: Data rate under varying channel conditions.

stop probing at that rate in the current round.

V. Conclusion

Rate adaptation algorithms are extremely important for WLANs with multirate capabilities. Previously proposed solutions are based on either statistics of ACKs, retransmissions, or signal strength measurements. In this paper, we have proposed and evaluated a novel sender-based ARF-like algorithm, PRA, which combines statistic-based methods with signal-measurement-based schemes to get the best of both worlds. It does not require any changes to the IEEE 802.11 standard. The novelty of our rate adaptation scheme lies in its great adaptability to a variety of channel conditions robust enough to be easily adopted for current wireless hardware. This rate adaptation framework also provides hints for systems (such as MIMO) used in future high-speed WLANs.

References

[1] IEEE Std 802.11-1999, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, Std., Aug. 1999.

[2] IEEE Std 802.11a-1999, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High-speed Physical Layer in the 5 GHz Band*, Std., Sept. 1999.

[3] S. Pal, S. R. Kundu, K. Basu, and S. K. Das, "IEEE 802.11 Rate Control Algorithms: Experimentation and Performance Evaluation in Infrastructure Mode," in *Proc. PAM'06*, Adelaide, Australia, Mar. 2006.

[4] Y. Yang, M. Marina, and R. Bagrodia, "Experimental Evaluation of Application Performance with 802.11 PHY Rate Adaptation Mechanisms in Diverse Environments," in *Proc. WCNC'06*, Las Vegas, USA, Apr. 2006.

[5] A. van der Vegt, "Auto Rate Fallback Algorithm for the IEEE 802.11a Standard," Utrecht University, Tech. Rep., 2002.

[6] P. Chevillat, J. Jelitto, A. N. Barreto, and H. L. Truong, "A Dynamic Link Adaptation Algorithm for IEEE 802.11a Wireless LANs," in *Proc. IEEE International Conference on Communications, 2003 (ICC'03)*, May 2003, pp. 1141–1145.

[7] M. Lacage, M. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: A Practical Approach," in *Proc. MSWiM'04*, Venice, Oct. 2004.

[8] "MADWIFI." [Online]. Available: http://madwifi.org/

[9] J. C. Bicket, "Bit-rate Selection in Wireless Networks," Master's thesis, Feb. 2005.

[10] G. Holland, N. Vaidya, and P. Bahl, "A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks," in *Proc. ACM MOBICOM'01*, Rome, Italy, July 2001, pp. 236–251.

[11] D. Qiao, S. Choi, and K. G. Shin, "Goodput Analysis and Link Adaptation for IEEE 802.11a Wireless LANs," *IEEE Trans. on Mobile Computing*, vol. 1, no. 4, October-December 2002.

[12] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, "OAR: An Opportunistic Auto-Rate Media Access Protocol for Ad Hoc Networks," in *Proc. ACM MOBICOM'02*, Atlanta, GA, Sept. 2002.

[13] J. del Prado Pavon and S. Choi, "Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement," in *Proc. IEEE International Conference on Communications 2003 (ICC'03)*, vol. 2, Anchorage, Alaska, May 2003, pp. 1108–1113.

[14] "ATHEROS Communications." [Online]. Available: http://www.atheros.com/pt/index.html/

[15] Netperf, "The Networking Benchmark." [Online]. Available: http://www.netperf.org/

1-4244-0357-X/06/$20.00 ©2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.