On Selecting the Best Transmission Mode for WiFi Devices

Mohammad Hossein Manshaei, Mathieu Lacage, Ceilidh Hoffmann, and Thierry Turletti
PLANÈTE Project-Team, INRIA, 06902 Sophia-Antipolis, France
Emails: hossein.manshaei@epfl.ch, {lacage,turletti}@sophia.inria.fr, cth@alum.mit.edu

***Abstract*— The design of efficient IEEE 802.11 physical rate adaptation algorithms is a challenging research topic and usually the issues surrounding their implementations on real 802.11 devices are not disclosed. In this paper, we identify and evaluate the key parameters to design such algorithms. We then present a survey on existing physical rate adaptation mechanisms and discuss their advantages and drawbacks. We also propose three new 802.11 physical rate adaptation mechanisms, named AARF, CLARA, and AMRR. AARF, proposed for low-latency systems, has low complexity and obtains similar performance than RBAR in stationary and non-fading wireless channels. CLARA is a culmination of the best attributes of the transmitter-based ARF and RBAR control mechanisms with additional practical features such as adaptive fragmentation to improve multipath fading channel sensing and to provide feedback control signalling. AMRR is designed for high-latency systems; it has been implemented and evaluated on an AR5212-based device. Experimentation results show more than 20% performance improvement in throughput over the default algorithm implemented in the AR5212 MADWIFI driver.**

I. Introduction

IEEE 802.11 is the de facto standard for WLANs and it is likely to play an important role in the next generation of wireless and mobile communication systems. The support of multi-rate [1] is obtained by employing different sets of modulation and channel coding schemes. The IEEE 802.11a/b/g WLAN working groups have defined the minimal requirements for physical (PHY) and medium access control (MAC) layer functionalities, but the exact transceiver architecture and the rate adaptation mechanism have been left open to WLAN device manufacturers. In the past few years, many rate adaptation mechanisms have been proposed in the literature. Selecting the most efficient mechanism is considered to be a challenging problem for WLAN devices. In fact, each rate selection mechanism has its own goal and improves the performance of 802.11 WLANs for a specific circumstance.

Mohammad Hossein Manshaei, the corresponding author, is now with LCA group at EPFL. Ceilidh Hoffmann was with Planete group as postdoctoral fellow.

Our contributions in this paper are summarized as follows. First, we identify the key parameters that have to be considered when designing efficient rate selection mechanisms. For instance, most of the previous work address the automatic rate adaptation without taking into account the implementation issues, such as the latency between MAC and PHY layers in existing 802.11 devices (see Section II-B). When such a parameter is considered, the effectiveness of these approaches are likely to suffer. The identification of these key parameters is definitely missing in the literature.

Second, we provide a review of the main rate selection mechanisms proposed for IEEE 802.11 devices, taking into account the important parameters discussed in this paper. For each mechanism, we provide the basic idea of the algorithm and discuss their pros and cons. Third, we describe three novel rate adaptation mechanisms, namely *adaptive auto rate fallback* (AARF), *adaptive multi rate retry* (AMRR), and *closed loop adaptive rate allocation* (CLARA). AARF and AMRR are two simple novel algorithms designed for one of the two classes of devices called high- and low-latency devices, respectively. The performance of AARF is close to the optimum represented by the theoretical RBAR [2] in the case of infrastructure networks and AWGN channel (non-fading channel). Our third proposed mechanism CLARA allows to better sense the wireless channel conditions and in particular to keep track of the multipath fading component. These mechanisms are evaluated using both experimentations and simulations done either by MATLAB or by the ns-3 [3] simulation tool. It is worth mentioning that AMRR has been integrated as a part of the MADWIFI driver [4].

Finally, this paper comes with a simulation package (i.e., WiFi module of ns-3 and simulation scripts) available in the public domain that allows for the reproduction of performance results presented. We believe that such a simulation package is very useful for the academic and industrial network research communities by making easier future

research on this area[1].

The remainder of the paper is organized as follows. In Section II, we provide the key parameters that have to be considered while designing efficient IEEE 802.11 rate selection mechanisms. In Section III, we present a review of current rate selection algorithms considering the key parameters identified in the previous section. Then, in Section IV, we describe AARF, AMRR and CLARA and evaluate them in Section V using both simulations and experimentations. Finally, we conclude in Section VI.

II. Key Parameters for Physical Rate Adaptation Mechanisms

This section presents the key parameters that should be taken into account when designing, implementing, and evaluating physical rate adaptation mechanisms. We first discuss the main features of wireless channels, such as fading. We then identify two classes of 802.11 devices namely *low-latency* and *high-latency* systems. In summary, low-latency systems allow for the implementation of per-packet adaptation algorithms and high-latency systems require periodic analysis of the transmission characteristics and updates to the transmission parameters. The classes of devices are the most important parameters, but they are usually ignored in most of rate adaptation mechanisms proposed. Finally we discuss why some algorithms have focused on time/throughput fairness, joint power/rate adaptation, or joint frequency/rate adaptation.

A. Wireless Channel Characteristics

Generally, the performance of communication systems over wireless channels is captured and analyzed using the BER as a function of Signal-to-Noise Ratio (SNR). Basically, BER decreases with large SNR observed at the receiver. However, the actual performance of the wireless system depends on several implementation issues and wireless channel characteristics. In the following we address these issues with a simple example in 802.11b. The same observations can be made with other transmission protocols such as 802.11a and 802.11g. In this example we present the BER calculation for 1 and 2 Mbps transmission rate in IEEE 802.11b WLANs.

According to the standard specification, the transmit signal is differentially encoded and BPSK/QPSK modulated for these transmission modes. The receiver can detect the signal coherently or differentially as well. In the latter approach it is not required to lock and track the carrier phase absolutely. If the signal is coherently detected, which is the case in this example, we denote them as DE-BPSK (i.e, *differentially encoded BPSK*) and DE-QPSK. It can be shown that in AWGN with Gray mapping, the probability of bit error for these two modulations (DE-BPSK and DE-QPSK) is $2Q\Big(\sqrt{2\frac{\mathcal{E}_b}{N_o}}\Big) - 2Q^2\Big(\sqrt{2\frac{\mathcal{E}_b}{N_o}}\Big)$ [5], [6], where $\frac{\mathcal{E}_b}{N_o}$ (or γ_b) is the average signal-to-noise ratio per bit. γ_b is equal to $SNR \times (BW/R_b)$, where R_b is the maximum bit rate of the modulation scheme (1 or 2 Mbps) and BW is the bandwidth of the signal (2 MHz for 802.11b). With maximum-ratio diversity combining, the fading-averaged BER for DE-BPSK and DE-QPSK can be obtained by Equation (1) [7], where $T_{il} = \frac{\binom{2l}{l}}{\binom{2(l-i)}{l-i}\{4^i[2(l-i)+1]\}}$, $\mu_c = \sqrt{\frac{\overline{\gamma}_b}{1+\overline{\gamma}_b}}$, $\bar{\gamma}_b$ is the average signal-to-noise ratio per bit, and L is the number of paths.

$$
\begin{aligned}
P_b \;=\; & \Big[\frac{1-\mu_c}{2}\Big]^L \sum_{k=0}^{L-1} \binom{L-1+k}{k} \Big[\frac{1+\mu_c}{2}\Big]^k - \Big\{\frac{1}{4} \\
& - \frac{\mu_c}{\pi}\Big[(\frac{\pi}{2} - \tan^{-1}\mu_c) \sum_{l=0}^{L-1} \frac{\binom{2l}{l}}{4(1+\overline{\gamma}_b)^l} - \sin(\\
& \tan^{-1}\mu_c) \sum_{l=1}^{L-1} \sum_{i=1}^{l} \frac{T_{il}}{(1+\overline{\gamma}_b)} [\cos(\tan^{-1}\mu_c)]\Big]\Big\} \quad (1)
\end{aligned}
$$

We can obtain similar results for Rician channel and non-coherent detection as well. Fig. 1 shows the results of the probability of bit error for these transmission rates using one or two paths (antennas) at the receiver side. We also plot the BER for AWGN channel ($L = \infty$). The results show that for a target BER as for example 10^{-5}, it is rather difficult to select a suitable SNR for the transmission rates without taking into account the number of receiver antennas, the type of diversity combining and the severity of multipath fading. This selection will be more difficult when we consider other PHY layer implementation issues like demodulator/detector (i.e., differentially or coherently) and *forward error correction* (FEC) codes in IEEE 802.11a/g.

In summary, the above example shows that an efficient rate adaptation algorithm should take into account the receiver structure (e.g., the number of receiver's antennas) and the wireless channel conditions for each packet transmission. For example, in public areas like airports, the wireless channel usually suffers from a deep fading caused by moving objects/people whereas in a small and calm office, the channel suffers from very short time changes. In summary, the rate adaptation mechanisms should behave in different manners for different devices and wireless conditions. As we will show in the following related work section, most of rate selection

[1]The simulation codes used in this paper are publicly available at: http://www-sop.inria.fr/planete/software

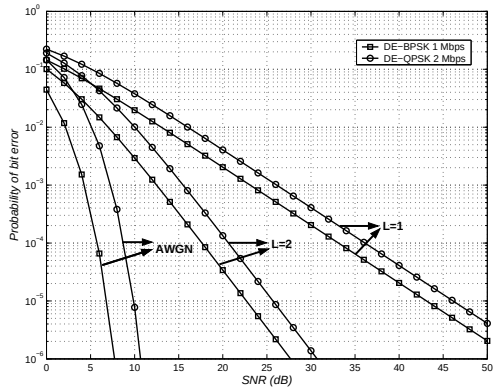

Fig. 1. BER as a function of SNR for 1 Mbps and 2 Mbps transmission rates in 802.11b.

mechanisms proposed do not consider the above issues and more often transmit packets based on a pre-computed table over an AWGN channel.

B. Implementation Issues: Low and High Communication Latency Systems

Typically, all 802.11 systems contain at least the following two subsystems: *the 802.11 PHY layer (or radio)* and *the 802.11 MAC layer*. The 802.11 radio integrates the modulation, demodulation, encoding, decoding, analog-to-digital converter, digital-to-analog converter, convolutional coder and decoder, and filtering functions. These functions are usually entirely implemented in hardware. The MAC layer is always implemented by a combination of dedicated hardware and dedicated software. The exact split between these two domains is entirely device specific.

The rate adaptation mechanisms we are interested in are part of the MAC layer. Their function is to choose the rate to be used for each packet that is sent to the PHY layer. The exact architecture of the MAC layer (i.e., how much of the MAC layer is implemented in hardware) varies a lot from one device to another. Therefore, it is very hard to design a device independent rate adaptation mechanism. However, it is clear that the *communication latency* between the PHY layer and the block that implements the rate adaptation mechanism within the MAC layer is one of the most important parameters to take into account when designing the mechanism.

Low-latency systems make possible the implementation of per-packet adaptation. This means that for each packet sent, feedback information on the transmission status of this packet is required before sending the next packet. As such, we calculate below the minimum time interval between two successive packet transmissions during a fragment burst ($T_{fragment}$), when a packet transmission fails ($T_{failure}$) and when a packet transmission succeeds ($T_{success}$). In the case of transmission failure, because the required ACK has not been received, the sending device starts a backoff procedure after a $DIFS$ at the end of $ACKTimeout$. The transmission of other packets by this device does not begin until the end of the backoff procedure, which cannot occur until $T_{failure} = ACKTimeout + DIFS + aCW_{rnd} \times aSlotTime$ after the end of the transmission, where aCW_{rnd} is a uniformly distributed variable between zero and aCW ($aCW_{min} < aCW < aCW_{max}$). In the case of successful transmission, the sending device either starts a $SIFS$ timer if it wants to keep on bursting the following MAC fragments ($T_{fragment} = SIFS$) or it starts a backoff procedure after a $DIFS$ if it wants to transmit another packet that is not a MAC fragment, i.e., $T_{success} = DIFS + aCW_{rnd} \times aSlotTime$. Because we are interested in the worst case scenario corresponding to the minimum time interval between two transmissions, we assume that $aCW_{rnd} = 0$ which gives; $T_{fragment} = SIFS$, $T_{success} = DIFS$, and $T_{failure} = ACKTimeout + DIFS$. And since $SIFS < DIFS$,

$$T_{fragment} < T_{success} < T_{failure}. \qquad (2)$$

The values of these parameters[2] are shown in Table I. If we do not allow the device to use a different rate for each fragment of a burst, the minimum latency requirement for all the values presented in Table I column $T_{success}$ is $28\mu s$. And if the user can change the rate for each fragment, e.g. using the new information from each fragment's ACK, the minimum latency requirement is $10\mu s$. In summary, any multi-standard system where the two-way communication latency between the PHY layer (where the transmission status is known) and the rate adaptation mechanism (where the information on the transmission status is acted upon) is higher than $28\mu s$ ($10\mu s$) cannot implement per-packet (per-fragment) rate adaptation.

The WaveLAN 802.11b Chipset [8] that employs an embedded processor and a dedicated communication bus with the baseband controller can be categorized as a low-latency system. More examples for low and high-latency wireless systems can be found in [9]. As a result, several rate adaptation mechanisms cannot be implemented in many existing devices, due to the latency characteristics of the

[2]Note that according to IEEE 802.11g standard specification, 802.11g devices can use long and short slot time, i.e., $20\mu s$ or $9\mu s$. Hence, $T_{success}$ could be equal to $50\mu s$ or $28\mu s$ in long and short slot time implementations, respectively. Since we are interested in minimum values, $T_{success} = 28\mu s$ has been considered here.

TABLE I
COMMUNICATION LATENCY CONSTRAINTS IN THE 802.11 STANDARDS

Standard	$T_{fragment}$	$T_{success}$
802.11 DSSS	10 μs	50 μs
802.11a	16 μs	34 μs
802.11b	10 μs	50 μs
802.11g	10 μs	28 μs

device. One of our contributions in this paper is the development of a new mechanism considering the high-latency system requirements.

C. Time and Throughput Fairness

The IEEE 802.11 WLANs support multiple data transmission rates using different sets of modulation and FEC. Although these data rates have increased considerably during past years from 1 Mbps to 54 Mbps, the MAC layer remains practically the same despite all proposed solutions in the literature. The only exception is the new IEEE 802.11e standard that allows the support of several service differentiation mechanisms IEEE 802.11e. However, it does not solve the "anomaly problem" common to CSMA/CA based mechanisms such as DCF. When wireless stations (STAs) with high PHY rates share a same wireless channel with one (or more) STA(s) with low PHY rate, the throughput performance of high PHY rate STAs[3] is considerably degraded [10]. The reason is that CSMA/CA mechanism aims to provide an equal opportunity to access the channel in the long term, and slow STAs capture the channel for a long time. Indeed, DCF provides *throughput fairness* while STAs with higher data rates at physical layer desire higher throughput at MAC layer (i.e., *time fairness*). The time fairness issue can be partially solved by selecting the best transmission rate and by offering more opportunities to the STAs with higher transmission rates. These techniques will be discussed in further detail in Section III.

D. Power Control and Frequency Allocation versus Rate Adaptation

Transmit power control (TPC) and *rate adaptation mechanisms* are the two most effective ways to achieve economical power consumption. Recently some efforts have been made for joint rate/power control in the IEEE 802.11 standard. Such mechanisms can use the new structures provided by the new standard specifications such as IEEE 802.11h to support intelligent TPCs and rate adaptation schemes simultaneously. In Section III, we will address a joint power/rate adaptation algorithm [11].

Besides multiple data rates, multiple channels are available for use in the IEEE 802.11 standard. Recently some research work have focused on frequency allocation in IEEE 802.11 wireless LANs. Particularly in these algorithms, if the channel conditions on a selected frequency channel are not favorable, STAs can skip to a better quality channel to send data with higher data rates [12]. Rate selection mechanisms that employ frequency allocation schemes to provide better performance are out of the scope of this paper.

III. PHYSICAL RATE ADAPTATION MECHANISMS:RELATED WORK

This section provides a detailed review and classification of rate adaptation mechanisms that have been proposed for IEEE 802.11 WLANs. To the best of our knowledge, this is the first detailed review and classification of the proposed rate selection mechanisms. For each mechanism, we summarize the key ideas and present their advantages and drawbacks. In this section, we present the mechanisms in the chronological order.

• ARF [13] was the first rate adaptation algorithm to be published in 1997. In ARF, each sender attempts to use a higher transmission rate after a fixed number of successful transmissions (i.e., 10) at a given rate and switches back to a lower rate after 1 or 2 consecutive failures. Simulations and experimental results show that ARF outperforms other rate selection algorithms when the channel conditions are suitable to send with the highest data rate [2], [14]. This scheme suffers from two problems. First, if the channel conditions change very quickly, it cannot adapt effectively. Second, the ARF mechanism will try to use a higher rate every 10 successfully transmitted packets whatever the wireless channel conditions (stable or not). This results in increased retransmission attempts and thus decreases the application throughput.

• RBAR [2] uses the RTS/CTS handshake mechanism to send back to the source the wireless channel conditions. The RTS, CTS and data frames are modified to include information on the size and rate of the data transmission in order to allow all the nodes within transmission range, at both the receiver's and sender's side, to correctly update their *network allocation vectors* (NAV). However RBAR has several flaws. It requires incompatible changes to the 802.11 standard that prevent its deployment in

[3]With the assumption that all STAs have the same size of frames, non-empty queues and do not experience any loss caused by transmission failure.

existing 802.11 networks. The threshold mechanism used in each receiver to select the best possible rate requires a calculation of the SNR thresholds based on an *a priori* channel model. Furthermore, RBAR has not considered the wireless channel parameters that are discussed in Section II-A. The mechanism assumes that the SNR of a given packet is available at the receiver, which is not generally true: some (but not all) 802.11 devices provide an estimation of the SNR by measuring the energy level prior to the beginning of the reception of a packet and during the reception of the packet. The use of RTS/CTS is mandatory in RBAR even though no hidden nodes are present. This can be a major performance problem.

• OAR [15] is a possible enhancement option to any automatic rate adaptation mechanism to provide *time fairness* (see Section II-C for further explanation). The key idea of OAR is to send multiple back-to-back data packets whenever the channel quality is good. But it suffers from several flaws. When channel conditions significantly vary during burst transmission, additional channel probing and RSH (*reservation sub-header*) overhead are required to inform others about changing the rates during burst transmission. It is not standard compliant as well and it requires extra modifications in the fragmentation protocol of the 802.11 standard to be integrated in existing devices.

• LA [16] is the first rate adaptation mechanism that uses the *received signal strength indication* (RSSI) measured from the AP frames to find the best transmission rate. With LA, STAs use the measured RSSI of all packets sent by AP that are addressed to themselves or the broadcast/multicast frames sent by the AP to calculate an average RSSI. The details of the LA mechanism have not been released publicly. Only some simulation results are provided, thus it is not clear how this algorithm behaves in practice and hence it is difficult to evaluate its performance.

• MiSer [11] is an algorithm based on the 802.11a/h standards whose goal is to perform joint rate/power control. The set of optimal rate/transmission power pairs is calculated offline with a specific wireless channel model. At runtime, STAs execute simple table lookups to choose the optimum rate/transmission power combination. MiSer suffers from the following flaws. It mandates the use of the RTS/CTS protocol. It requires the choice of an *a priori* wireless channel model for the offline table calculation. Thus it is not a practical approach considering the implementation issues addressed in Section II-A.

• MAD [17] obtains the instantaneous physical channel conditions information from several receivers and then decides to send the packet to the STA that maximizes the total throughput of the network. The analytical and simulation results presented in [17] show that MAD can improve the overall throughput of network by 50% compared to OAR. However, MAD inherits all the drawbacks of RBAR and OAR rate adaptation mechanisms. The definition of the new RTS packet type (i.e., Group-RTS), as well as the new frame format for CTS and DATA packets, are not standard compliant. Thus, MAD cannot be implemented in current WLAN devices.

• MADWIFI [4], [18], [19], designed for AR5212 chipset, is one of the most widely used implemented physical rate adaptation mechanisms. It allows the user to create up to 9 unbounded FIFOs of transmission descriptors to schedule packets for transmission. Each descriptor contains a status field that holds the transmission status of the descriptor, a pointer, and the size of the data to be transferred. Each transmission descriptor also contains an ordered set of 4 pairs of rate and transmission count fields (r_0/c_0, r_1/c_1, r_2/c_2, r_3/c_3). Whenever the wireless medium is available for transmission, the hardware triggers the transmission of the descriptor located at the head of the FIFO. If this transmission fails, the hardware keeps on trying to send the data with the rate r_0, $c_0 - 1$ times. If the transmission keeps on failing, the hardware tries the rate r_1, c_1 times then the rate r_2, c_2 times and finally the rate r_3, c_3 times. When the transmission has failed ($c_0 + c_1 + c_2 + c_3$) times, the transmission is abandoned: the status field of the descriptor is updated and it is transferred back from the local RAM to the system RAM. In MADWIFI, the short-term variations are handled by the multi rate retry mechanism and the long-term variations are handled by changing the value of the r_0/c_0, r_1/c_1, r_2/c_2 and r_3/c_3 pairs at regular fixed intervals (from 0.5 to 1 second intervals). We will discuss the performance evaluation of this mechanism in Section V-C, when we introduce our practical approach using this mechanism.

• SampleRate [18], [19] sends packets at different data rates periodically to gather the information about other data rates. Then it selects the data rate that obtains the highest throughput. This mechanism suffers from a high number of packet losses, although it can provide higher throughput for certain network conditions.

• HRC [20], [21] is the first rate adaptation mechanism that makes a differentiation between *short-term* and *long-term* variations of the chan-

nel conditions. Basically, the core of HRC is a throughput-based rate controller. It probes adjacent rates to determine if a rate change is necessary. In addition, based on the information received from the rapid *signal strength indication* change detector module in HRC, the lookup table uses two different sets of thresholds, named stable and volatile low thresholds. HRC achieves a much higher quality than static-based algorithms when used to send real-time video streaming over 802.11a wireless LANs. The performance of this mechanism has only been compared with the algorithm implemented for the R5000 chipset by Atheros.

- FAR [22] is an enhanced version of RBAR. It is based on two key ideas: First, it adapts the transmission rate for RTS/CTS/ACK frames while other mechanisms suppose that all control packets should be sent with the basic rate. Second, in FAR, the rate of RTS/ACK frames is selected at the sender side while it is done at the receiver side for DATA/CTS frames. But FAR suffers from three important flaws. First, the RTS/CTS mechanism is mandatory even though no hidden nodes are present. Second, the mechanism proposed to update the NAV should be implemented for all the STAs in the network. Thus FAR cannot be deployed easily in existing 802.11 WLANs. In other words, FAR devices cannot coexist with the 802.11 compliant devices without employing *modified virtual carrier sensing* (MVCS). Furthermore, further investigation on MVCS is required to address the possible usage of fragmentation.

- It is also useful to mention several analytical research work on rate adaptation algorithms. In [14] and [23], Qiao et al. present a complete analytical evaluation of throughput for link adaptation in IEEE 802.11a taking into account the transmission modes and the frame length. In [24], Wu et al. propose and evaluate the usage of the SNR parameter to select the best transmission rate. They have modified the MAC header and reservation scheme because of the multi-rate hidden terminal problem. Finally in [25], Pang et al. propose an automatic rate fallback mechanism that aims to achieve packet loss differentiation (i.e., transmission errors vs. collisions).

To summarize, we have classified all rate adaptation mechanisms in Table II according to their characteristics and the key parameters they consider. We can note that each rate adaptation aims to improve the performance of WLANs considering one or several key parameters.

IV. Our Approach to Physical Data Rate Adaptation

Before describing in detail the AARF, CLARA and AMRR rate adaptation mechanisms, we present our motivation and how they relate with other proposals.

As we have just discussed, most of rate adaptation mechanisms proposed (MAD, FAR, Miser, HRC, and OAR) select the best data rate using pre-computed tables based on an AWGN channel. The only mechanism that tries to update the threshold values is HRC, while the other mechanisms use a simple static AWGN channel model. HRC and SampleRate are the only ones that try to predict the wireless channel conditions. Most of the mechanisms fail to consider the device dependent (e.g., number of antennas) and wireless channel characteristics (e.g., fading), as discussed in Section II. This motivated us to design the CLARA mechanism to better sense the wireless channel characteristics.

Regarding the implementation issues presented in Section II-B: It is almost impossible to implement the mechanisms such as RBAR and OAR because they need higher processing time than the one available in communication latency in existing wireless devices. In contrary, the three approaches we propose in this paper are standard compliant and consider implementation issues such as latency.

As shown in Table II, OAR, MAD, and MOAR provide time fairness using the techniques described in Section II-C and III. Typically, OAR and MAD use a simple back-to-back burst transmission mode in that the sender is allowed to send several frames back-to-back in a burst, provided that the entire frame exchange duration does not exceed a threshold value (e.g, proportion of transmission rate over basic rate). MAD proposes a packet concatenation (PAC) mechanism. Such an algorithm can be integrated in all rate adaptation mechanisms to provide time fairness. For example, OAR is already integrated in RBAR. Note that there are some schemes such as burst transmission mode that are being standardized as a part of 802.11e MAC [26]. Such algorithms can simply be integrated in other mechanisms too.

As explained in Section II-D, the mechanisms that adapt the transmission parameters to the channel conditions can be designed to optimize power consumption and/or throughput. As we focus on the task of maximizing the application-level throughput through rate adaptation mechanisms, we do not use any TPC mechanism in our proposed approaches. Regarding joint frequency/rate adaptation, *multi-band opportunistic auto rate* (MOAR) is the only

TABLE II
COMPARISON BETWEEN RATE ADAPTATION MECHANISMS

Adaptation Mechanisms	*Feedback Based*	*Standard Compliant*	*Prior Table Calculation*	*Power Control*	*Need Low Latency*	*Time Fairness*	*Key ideas*
AARF	✓	✓	–	–	✓	–	*Low-latency system/Improve ARF*
AMRR	✓	✓	–	–	–	–	*High latency systems*
ARF	–	✓	–	–	✓	–	*Easy to implement*
CLARA	✓	✓	✓	–	✓	–	*Sensing the wireless channel/Use fragmentation*
FAR	✓	✓	✓	–	✓	–	*Enhanced RBAR to be implemented*
HRC	–	✓	✓	–	–	–	*Differentiation of long/short channel modification*
LA	–	✓	–	–	–	–	*Using RSSI*
MAD	✓	–	✓	–	✓	✓	*Group feedback/provide time fairness*
MiSer	–	–	✓	✓	✓	–	*Joint power/rate adaptation*
MOAR	✓	–	✓	–	✓	✓	*Joint Frequency/rate adaptation in adhoc*
OAR	✓	–	✓	–	✓	✓	*Provide time fairness*
RBAR	✓	–	✓	–	✓	–	*Get receiver feedback by RTS/CTS*
SampleRate	–	✓	–	–	–	–	*Send with high rates/accept packet losses*

proposed algorithm. MOAR uses an optimal band skipping rule to find the best band for transmission every time a node pair gains access to the medium. This could be an open issue for future work on IEEE 802.11 PHY layer rate adaptation.

In summary, our approach to physical data rate adaptation contains three major parts. We first propose *adaptive ARF* (AARF), an extension to ARF that improves its performance in presence of stable channel conditions. Then we propose the CLARA mechanism that aims to make easier multipath fading sensing and feedback control signaling. Finally, we present *adaptive multi rate retry* (AMRR), a practical mechanism proposed for AR5212-based devices.

A. Adaptive Auto Rate Fallback: AARF

ARF was designed for low-latency systems based on the second generation of WaveLAN devices. Although this scheme can cope with short-term variations of the wireless medium characteristics, it fails to handle stable conditions. Typically, office workers setup their laptop, sit in a chair or at their desk and work there for a few hours. ARF can recognize this best rate and use it extensively but it also keeps trying to use a higher rate, every 10 successfully transmitted consecutive packets, to be able to react to possible channel conditions changes. This process however can be costly because periodic transmission failures generated by ARF decrease the application throughput. ARF cannot provide a stable PHY rate for a long period of time because it employs the same strategy for long-term and short-term variations of the wireless medium characteristics. To avoid the scenario described above, an obvious solution is to increase the threshold used to decide when to increase the current rate from 10 to 40 or 80. This approach can indeed improve performance in certain scenarios, but it does not work in practice as it completely disables the ability of ARF to react to short-term channel conditions changes.

This problem led us to the key idea of AARF. In AARF, the threshold is changed at runtime to better reflect the channel conditions. This adaptation mechanism increases the amount of history available to the algorithm, which helps it to make better decisions. In AARF, we chose to adapt this threshold by using a *binary exponential backoff* (BEB), introduced in [27]. When the transmission of the probing packet fails, AARF switches back immediately to the previous lower rate (as in ARF) but also multiplies by two the number of consecutive successful transmissions (with a maximum bound set to 50) required to switch to a higher rate. This threshold is reset to its initial value of 10 when the rate is decreased due to two consecutive failed transmissions. The pseudo code that describes formally the behavior of ARF and AARF is available in [9]. Basically, AARF increases the period between successive failed attempts to use a higher rate. Fewer failed transmissions and retransmissions improves the overall throughput. For example, Fig. 2 shows a period of time where the most efficient transmission mode is mode 3. ARF tries to use mode 4 after 10 successful transmissions with mode 3, whereas AARF uses the history of the channel and does not increase the rate at each 10 successful packet boundary.

B. Closed-Loop Adaptive Rate Allocation: CLARA

The key idea of CLARA is based on the following observations [28]:

- The frame loss in IEEE 802.11 results from MAC collision or PHY corruption.
- The cause of frame corruption can be differentiated to stable versus unstable channel state.

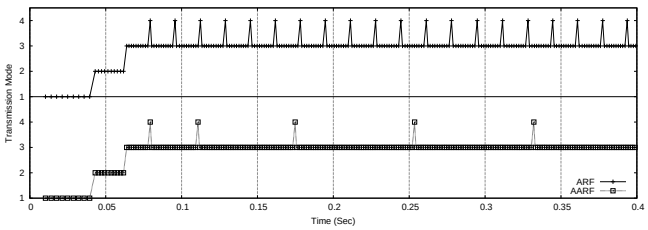

Fig. 2. Mode selection comparison between ARF and AARF.

- MAC collision can be eliminated using the optional RTS/CTS handshake mechanism.
- Channel stationarity can be estimated by partitioning data frames into many smaller fragments.

We first address the last observation. Basically, due to non-stationarity of the multipath-fading time-varying channel, bit errors occur in bursts during periods of *deep fade*. This is an example where a longer frame duration is more susceptible to data corruption due to fluctuation of the received signal strength. Typically, the stationarity of a channel is commonly measured in terms of its coherence time T_c. For the successful delivery of a MAC frame, we must consider the relative duration of the channel coherence time over the entire MAC frame duration T_f. Consider the example illustrated in Fig. 3, where the non-stationary channel has two different signal strengths in intervals $\Delta_1 = 0 \leq t \leq T_1$ and $\Delta_2 = T_1 \leq t \leq T_2$. In a unique situation where the initial and final time epochs of the data frame are located in different Δ intervals $T_i \in \Delta_1$ and $T_i + T_f \in \Delta_2$, two scenarios are possible. If a lower rate mode is selected such that the channel is stable for the entire frame duration, the frame is received error-free but at a sub-optimal transmission rate. But, if a higher rate mode is selected such that the channel is unstable, the frame is corrupted and must be retransmitted with a reduction in overall throughput. Throughput is maximized if DATA can be transmitted in two different modes.

Such a mode selection procedure can be achieved by the appropriate fragmentation of the data frame. As shown in Fig. 3, each fragment duration T_{frag} is carefully chosen to retain channel stability during the transmission of each fragment. Now we describe how all the above functionalities are incorporated into CLARA. With every connection, we propose the use of RTS/CTS protocol for initial hand-shake and channel sensing if the frame duration is above the RTS threshold. The RTS threshold depends on the number of nodes in the WLANs that are calculated in [28]. If the RTS/CTS handshake is successful, the channel is reserved for the entire

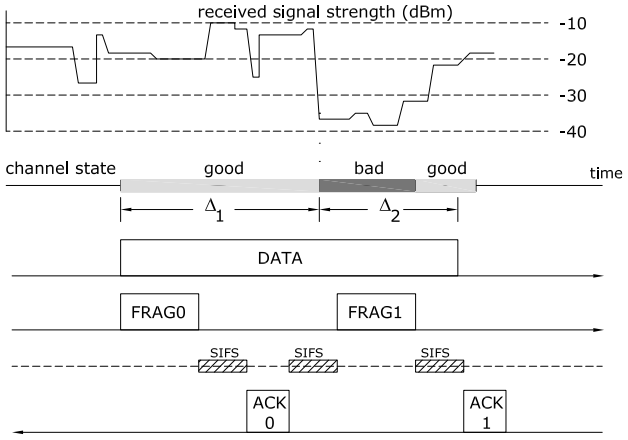

Fig. 3. DATA or fragment delivery over a non-stationary channel.

duration of the data frame.

Note that in RBAR, channel probing via RTS/CTS is mandatory regardless of the frame size or data rate. If an ACK is not received, it assumes the cause is a bad channel state (due to non-stationarity of the PHY channel) or poor mode selection (due to inaccurate receiver feedback measurement). However, it is not possible to differentiate one type of frame loss from the other in RBAR. This is because RTS/CTS serves only for initial probing of the PHY channel. During data transmission, the channel statistics may have changed and the initial estimate based on RTS reception may not be adequate.

In CLARA, RTS/CTS is used in the conventional sense of channel reservation and hidden node identification. Unlike with RBAR, channel statistics and SNR are not measured based on RTS reception. Therefore, the CTS duration/ID field need not be altered. The transmission mode for data delivery is selected a priori based on feedback data over previous CTS and/or ACK frames. In this respect, our rate selection method is similar to ARF as the history of past channel access statistics is used in rate selection. Unlike ARF, CLARA has a much broader knowledge of the channel statistics by using feedback reserved bits, as well as an increase in the number of ACKs due to fragmentation. Contrary to RBAR, there is no mandatory requirement for RTS/CTS handshake in our scheme.

It is important to mention that data fragmentation is optional in CLARA. Based on the RSSI history (similar to [16] and [21]) from previous fragment ACKs, the channel coherence time can be estimated. Unlike with the RTS/CTS handshake, fragments and their corresponding ACK frames are used for the sole purpose of PHY channel probing. As it is stated in the 802.11 standard [1], a data fragment and its

ACK frame serve the role of a virtual RTS/CTS. We should, however, be aware that fragmentation is used to probe PHY channel stability. It is more appropriate to view the combination of RTS/CTS and fragmentation as the creation of a virtual duplex PHY channel. Finally it should be noted that there are 5 and 9 reserved bits available in PLCP header of 802.11b and 802.11a modes respectively for carrying feedback information [28]. In practice, these bits will be used to carry RSSI and other relevant control information from the receiver side.

C. Adaptive Multi Rate Retry: AMRR

In this section we present our practical approach for improving the performance of MADWIFI. Although AARF has been originally designed for *low-latency* systems, the AR5212-based 802.11 device falls in the *high-latency* group. Basically, a natural way to introduce a binary exponential backoff in MADWIFI is to adapt the length of the period used to change the values of the rate/count pairs and this is exactly what *adaptive multi rate retry* (AMRR) does. To simplify the logic of the code, we also use simpler heuristics than those in MADWIFI to choose the rate/count pairs at the period boundaries.

To ensure that short-term variations of the wireless medium are quickly acted upon, we chose $c_0 = 1$, $c_1 = 1$, $c_2 = 1$ and $c_3 = 1$ (whereas MADWIFI uses $c_0 = 4$, $c_1 = 2$, $c_2 = 2$ and $c_3 = 2$, with c_i defined in Section III for the MADWIFI bullet). The rate r_3 is always chosen to be the minimum rate available (typically, 6Mbps in 802.11a networks). The rates r_1 and r_2 are determined by r_0: we implemented the simplest heuristic possible by setting r_1 and r_2 to the immediately lower available rates. Finally, AMRR updates r_0 from the previous value of r_0 and the transmission results for the elapsed period. The exact heuristics are detailed in the pseudo codes available in [9].

D. Discussion

The rate adaptation mechanisms proposed in this section have been designed considering several key parameters identified in Section II. In particular, AARF has been designed for non-fading wireless channels, whereas CLARA can detect the presence of multipath fading in wireless channel and select accordingly the best transmission rate. In order to obtain the benefits of these mechanisms simultaneously, a dynamic channel detection should be deployed in wireless devices to detect the channel conditions and select the best transmission rate based on the suitable mechanism, e.g., CLARA or AARF. Note that such a mechanism has already been implemented in HRC by monitoring the RSSI parameter [21].

V. Performance Evaluation of Proposed Approaches

In this section, we analyze the performance of our proposed mechanisms using both simulations and experimentations. We use the WiFi module of the ns-3 simulator to evaluate the performance of AARF, ARF, RBAR, and AMRR. This module implements an accurate model of the 802.11a PHY layer and of the 802.11 MAC DCF functionality. An overview of ns-3 and the available 802.11 PHY model implemented is available in [3]. ns-3 is covered by the GNU GPLv2 license and is publicly available for research, development, and use [3]. We also conduct some simulations in MATLAB to evaluate the performance of CLARA in both stationary and non-stationary wireless channels and compare it with ARF. Finally we present our experimental results for AMRR and MADWIFI.

A. Performance Evaluation of AARF

We compare the performance of AARF with ARF and RBAR in an infrastructure network, as ARF and AARF have been originally designed for this specific environment. Indeed, ARF and AARF decide to reduce their PHY transmission rate based on one or two packets, and this delay is usually higher than the channel coherence time of a dense wireless network environment with multiple connections. In this context, it is not possible for ARF or AARF to adapt to the channel characteristics correctly; instead, we proposed CLARA for such scenarios.

Our simulation scenario contains two stations. STA A remains static while STA B moves toward station A. The movement of STA B is not continuous: STA B stays static for one minute before moving 5 meters towards STA A. Whenever STA B stops, a single CBR data transmission towards STA A is started. Each CBR packet is 2000 bytes long. Because we need a saturated traffic, each CBR flow attempts to transmit at 60 Mbps for 270 second of simulation. The simulations have been performed by version *2909:30c9b48a3af3* of ns-3 simulator.

To analyze the influence of the AARF algorithm parameters, we ran a set of simple simulations that kept all parameters constant, except one. The default fixed values and the variation range of these parameters, are shown in Table III.

The detailed results of these simulations are shown in [29]. The TimerTimeout parameter, as defined by ARF, is used as an alternative for successful

TABLE III
DEFAULT AARF PARAMETERS

Parameter (unity)	*Default*	*Variation Range*
TimerTimeout (nb of packets)	*15*	*11-100*
MinSuccessThreshold (nb of packets)	*10*	*1-49*
MaxSuccessThreshold (nb of packets)	*50*	*11-100*
SuccessFactor (no unit)	*2*	*1.01-5*

packet transmission threshold (i.e., 10 packets), to increase the data rate. As detailed in [14], we used a packet-based timer for ARF and AARF rather than the time-based timer originally described in [13]. The authors of [2] and [14] had already established that its value had little influence on the behavior of ARF (be it time-based or packet-based) and Figs. 2 in [29] shows that it also has no noticeable influence on the behavior of AARF. The *MinSuccessThreshold* and *MaxSuccessThreshold* are the minimum and the maximum value of successful transmission threshold respectively. Fig. 3 in [29] shows that *MinSuccessThreshold* has no noticeable influence on the behavior of AARF. The *MaxSuccessThreshold* parameter has the highest influence on the performance of the algorithm (see Fig. 4). The performance of the algorithm reaches a plateau toward the values of 80 to 90. We chose 50 because this value offers a good tradeoff between the performance obtained and the ability of the algorithm to increase its rate within a reasonable amount of time when the user moves toward the AP. The *SuccessFactor* is the increasing factor for *MaxSuccessThreshold*, which has been set empirically to 2 (i.e. binary exponential) as it is shown in Fig. 4 of [29].

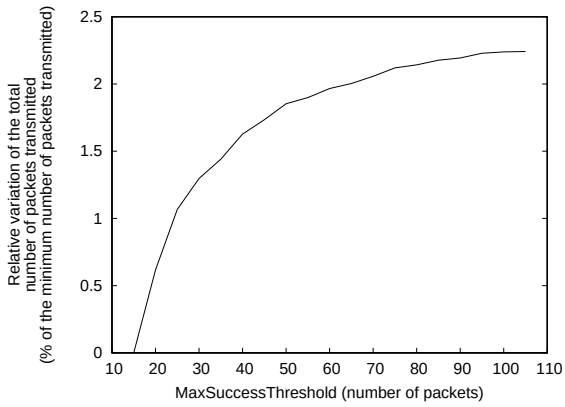

Fig. 4. Influence of the value of *MaxSuccessThreshold* on the performance of AARF.

Fig. 5 shows the mean goodput (the goodput represents the application throughput) achieved by ARF, AARF, and RBAR in the same conditions. In these simulations, PLCP headers and RTS/CTS/ACK frames are sent with BPSK modulation with a FEC rate equal to $1/2$ and a 6 Mbps data rate, which corresponds to the basic mode in 802.11a. Note also that all throughput shown in this paper exclude the MAC and PHY headers. These

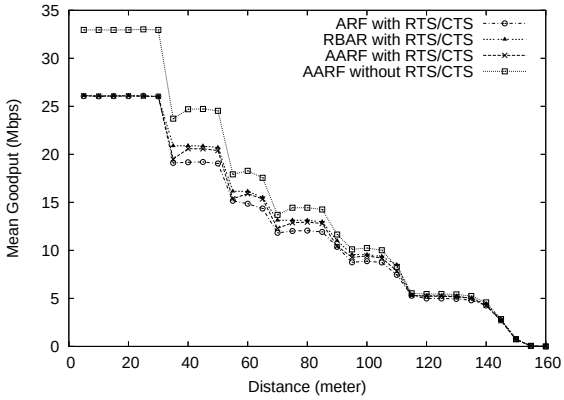

Fig. 5. Mean goodput for a single hop with three different automatic rate selection algorithms. The available data rates in our simulations are 6, 12, 18, 24, 36, and 54 Mbps.

results show that ARF fails to perform as well as RBAR for mode 12, 18, 24, and 36 Mbps. The main reason for this is explained in Section IV-A: ARF periodically causes transmission failures. RBAR always picks the best available rate, which means that the number of transmission failures is much lower. Its mean goodput is thus much higher. Fig. 5 shows that AARF performs on average the rate selection as well as RBAR and better than ARF. One of its main advantage over RBAR is that it does not require the use of the RTS/CTS option. As expected, in this case its performance is much higher than that achieved with RBAR as shown in Fig. 5.

Fig. 6 and 7 show the ARF and AARF average transmission delay and jitter obtained at the application layer for the simulated single hop with a voice over IP application. The data rate of the generated traffic is 8 Kbps, where the packet length is 200 Bytes. The transmission delay is calculated considering the maximal number of retransmissions, which is equal to 4. The results show that the average transmission delay of ARF is slightly higher than that of AARF. The reason is that AARF uses a more conservative strategy than ARF to select the data rate.

Note that in congested networks, where the number of collisions increases dramatically, both ARF and AARF can not increase the transmission rate (because to do so, at least 10 consecutive packets successfully received are required). In such scenarios, the RTS/CTS option is highly recommended to avoid the performance degradation [30]. Other

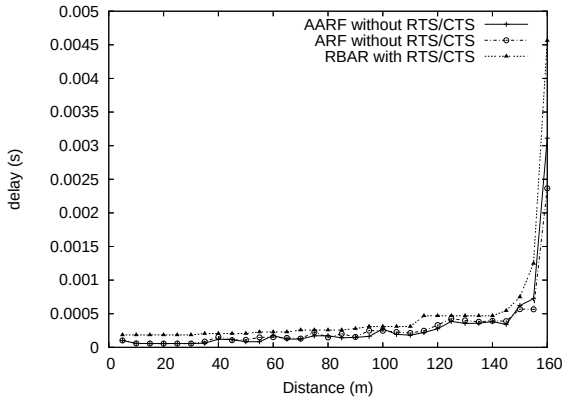

Fig. 6. Transmission delay of a single hop with ARF and AARF.

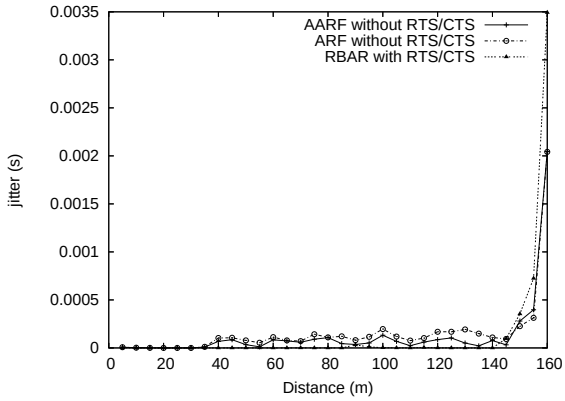

Fig. 7. Transmission delay of a single hop with ARF and AARF.

simulations (not included in the paper) show that with RTS/CTS transmission and in the presence of a lot of contending stations, AARF still outperforms ARF and that AARF can reach on average the near optimum performance of the RBAR algorithm.

Our simulation results clearly show that AARF outperforms ARF. In particular, AARF can reach on average the near-optimum performance of the RBAR algorithm without requiring any incompatible changes to the 802.11 protocol. Furthermore, all it requires from the hardware is a low communication latency between the block that implements the rate control algorithm and the transmission block that handles the ACK timeouts. This new algorithm can thus be easily and incrementally deployed in existing infrastructure networks with a simple firmware or driver upgrade on each node.

B. Performance Evaluation of CLARA

To evaluate the performance of CLARA we used event-driven simulations carried out in MATLAB. Here we compare the performance with ARF. We do not compare CLARA with RBAR because RBAR is a subset of CLARA[4]. In other words, our approach is a practical and improved RBAR as feedback information is piggy-backed through both CTS and fragment ACKs.

It should be noted that we use the Rayleigh distribution to model the severity of received signal amplitude, quantified in terms of the Rayleigh parameter σ (i.e. the mean value is $\sqrt{\pi/2}\ \sigma$ and the received signal power is chi-square distributed). The channel stationarity is modeled as a Poisson arrival process with arrival rate λ, i.e., a parameter equivalent to the channel coherence time T_c. As this study focuses on the PHY layer behavior, only two STAs are used in the system. The MAC frame size is set to 1500 bytes. If fragmented, the maximum number of fragments is 4. The STAs are operating in 802.11b mode with long PLCP preamble and are capable of communicating in all 11b modes. In each event 10,000 data frames or 40,000 fragments are exchanged. The sender selects the best mode based on real-time feedback information. The PHY transmission mode selected for RTS is the one used for the latest frame or fragment transmitted.

Fig. 8 shows the throughput performance of CLARA and ARF with and without fragmentation. As expected, CLARA outperforms ARF whatever the SNR range. For low SNR (5–15 dB), fragmentation is preferred. The gap between CLARA and ARF closes as the channel becomes more stationary (from 2 to 8 ms). The remaining gap is due to the incapability of ARF to adapt to channel fluctuations. For high SNR, fragmentation is not recommended as finer channel sensing is not required and the overhead loss of fragmentation reduces the throughput. Additionally, when the channel is more stationary there is little performance gain by fragmentation.

In order to show the benefits of using fragmentation in non-stationary channels, we run a series of simulations for T_c=0.4 and 10 ms as shown in Fig. 9. We note that fragmentation allows to better sense the channel and as a result, the throughput is not susceptible to deep channel fades and variations, resulting in better QoS in terms of smaller delay and steady data flow. In particular, ARF with its slow rate adaptation leads to choppy data flow and long delays. Note that for low received SNR, CLARA with fragments outperforms its cousin without fragmentation. This trend is reversed for either more stationary or less severe channel conditions. Regardless, fragmentation causes a smoother throughput in

[4]In particular, if the reserved bits in the PLCP header are used instead of changing the MAC, RTS, and CTS frame format, RBAR will become a subset of CLARA. Note that CLARA uses fragmentation as an option to sense the wireless channel conditions.

(a) (b)

Fig. 8. Plot of Throughput vs. received $\overline{\text{SNR}}$ with data fragmentation; (a) coherence time T_c=2ms, (b) 8 ms.

(a) (b)

Fig. 9. Plot of Throughput for each fading realization; (a) T_c=0.4 ms, $\overline{\text{SNR}}$=10 dB , (b) T_c=10 ms, $\overline{\text{SNR}}$=15 dB.

both ARF and CLARA.

C. Performance Evaluation of AMRR

To evaluate the performance of AMRR and compare it to that of MADWIFI and RBAR, we used the simulation environment described in Section V-A. The MADWIFI algorithm we simulated is a trivial copy of the code available in the MADWIFI driver, slightly modified for the simulation environment to use the 6 transmission modes chosen for our 802.11a networks. Our implementation of MADWIFI in the simulator and of AMRR both in the simulator and in the driver is straightforward except for the way the transmission FIFO, which is shared between the AR5212 chip and the Linux kernel driver, is handled.

More specifically, the original MADWIFI driver initialized the transmission descriptors present in the FIFO only once, when they were inserted into the FIFO. One of the consequence of this behavior is that it can generate wide oscillations of the algorithm due to the different rates of the packets located at the head and at the tail of the FIFO. For example, when the user application generates a 15 Mb/s data flow and if the wireless channel conditions allow the 802.11a 12Mb/s transmission mode with a reasonable PER ($r_3 = 12$, $r_2 = 6$, $r_1 = 6$ and $r_0 = 6$), the source buffers quickly fill (the transmission descriptor FIFO is thus full) and the user application encounters a lot of packet drops at the source. If the PER is low-enough at this rate set, the rate control algorithm will try to increase the rate set to $r_3 = 18$, $r_2 = 12$, $r_1 = 6$, and $r_0 = 6$, this means that every new packet that enters the FIFO uses this new rate set.

However, at the next decision period boundary, the transmission statistics used to adapt the current rate set are those generated by the transmission of the packets whose rate set is $r_3 = 12$, $r_2 = 6$, $r_1 = 6$, and $r_0 = 6$ and that are still present in the FIFO. Because the PER of this rate set is low enough, the rate control algorithm thus will try to increase the rate set again, yielding something like ($r_3 = 24$, $r_2 = 18$, $r_1 = 12$, and $r_0 = 6$).

At one point, the packets whose rate set is high will reach the front of the FIFO and will be treated by the hardware: They are likely to fail, which will make the rate control algorithm drop the current rate set quickly. However, it is likely to decrease the rate set too much for the same reasons it increased it too much previously. We observed this phenomenon during preliminary experiments and we reproduced it in a simulation as shown in the curve named MADWIFI driver in Fig. 10. To avoid this problem we have modified the MADWIFI driver to parse

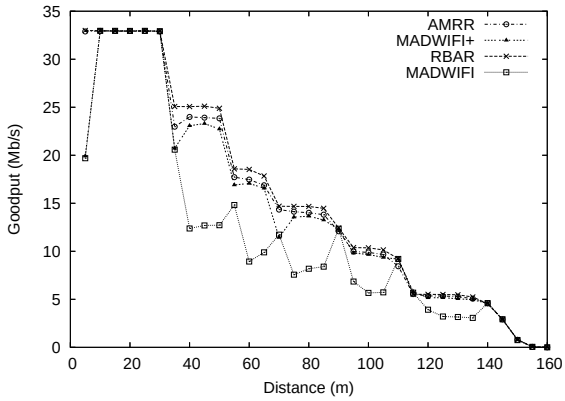

Fig. 10. Mean goodput for a single hop with RBAR, MADWIFI, MADWIFI+, and AMRR mode selection.

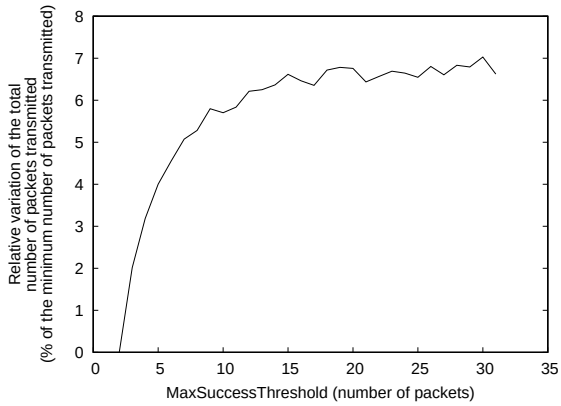

Fig. 11. Influence of the value of *MaxSuccessThreshold* on the performance of AMRR.

the transmission FIFO each time a rate change happens to apply the rate change to each transmission descriptor concerned immediately. All further simulations and experiments (unless explicitly stated) were conducted with this modified version of the MADWIFI algorithm named MADWIFI+.

Because AMRR is based on the same set of ideas developed for AARF, that is, the use of a BEB to adapt the success threshold, similar parameters can be tweaked. Among these, the *MinSuccessThreshold* and the *MaxSuccessThreshold* parameters are the two most important parameters. We did not evaluate the influence of *MinSuccessThreshold* because increasing it would further decrease the ability of the AMRR algorithm to react to channel conditions changes. Fig. 11 shows how *MaxSuccessThreshold* influences the performance of the AMRR algorithm.

As expected, *MaxSuccessThreshold* follows the same pattern observed in Fig. 4-d: the throughput increases with its increase. As in Section V-A, we do not choose the highest possible value to avoid decreasing its ability to react fast enough to channel conditions changes and set its value to 15, which is close to the plateau observed in Fig. 11. The simulation results shown in Fig. 10 confirm that AMRR performs much better than the original rate control algorithm used in the MADWIFI driver and that it achieves on average similar performance to RBAR. Here again, the BEB-based adaptive mechanism is the main reason for this throughput improvement: the probability of trying a rate set that requires numerous retransmissions is greatly reduced. We conducted some experimental evaluations for AMRR as well. Our test setup was created to approximate as closely as possible real-world use cases. As such, we chose a typical office environment with many people walking from one office to the other: a 802.11b/g AP (a Netgear WG602) was setup with a private access point in an office and a laptop with a *Proxim Orinoco Gold* pcmcia card based on the AR5212 chipset was setup in another office approximatively 10 meters away from the AP. We first installed an unmodified 2.6.5 Linux kernel and a RedHat 8.0 Linux distribution on the test laptop and then tested three versions of the MADWIFI driver:

- MADWIFI driver: the original unmodified MADWIFI driver.
- MADWIFI+: the MADWIFI driver modified to apply immediately rate changes on its transmission FIFO as described above in this section.
- AMRR: the MADWIFI driver modified to apply immediately rate changes and implement the AMRR rate control algorithm.

To mitigate the variations of the transmission conditions with time, we ran three sets of experiments whose results are shown in Fig. 12. The goal of each of the three sets of experiments was to compare the average throughput achieved by two of the three drivers. For each set of experiments, we loaded in the Linux kernel alternatively each of the two selected drivers and started a 10 minutes continuous 30 Mbps UDP stream from the laptop to the only machine located on the 100 Mbps ethernet link of the AP. We executed this test 5 times for each of the two selected drivers and recorded the average throughput achieved during each experiment.

Despite the variability of the experiments, we can observe the performance improvement achieved by AMRR over both MADWIFI and MADWIFI+ in Fig. 12. AMRR reached on average 24Mbps and both MADWIFI and MADWIFI+ reached on average 20Mbps. Fig. 12-c shows that MADWIFI and MADWIFI+ obtain roughly similar throughput, even though a clear throughput oscillation for the MADWIFI driver can be noticed during these experiments.

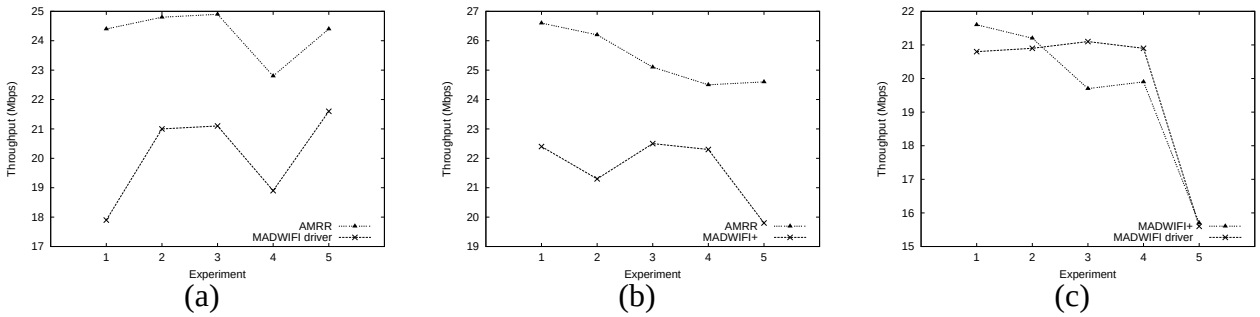

Fig. 12. Experimental results for: (a) AMRR versus MADWIFI driver, (b) AMRR versus MADWIFI+, (c) MADWIFI versus MADWIFI driver.

VI. Conclusion

In this paper, we identified the most important parameters that have to be considered when designing efficient rate selection mechanisms. We provided a review and classification of the main rate adaptation algorithms proposed for IEEE 802.11 devices. Then, we presented three novel mechanisms for rate adaptation. AARF and AMRR have been respectively designed for *low-latency* and *high-latency* communication systems. CLARA uses some practical features to facilitate multipath fading channel sensing and feedback control signaling to better sense the wireless channel conditions. As CLARA is implemented at the PHY layer, it is MAC independent and therefore, it can be implemented in all existing and emerging 802.11 WLAN standards. Finally, this paper comes with a simulation package released under the GPLv2 license that allows for the reproduction of all the performance results shown.

References

[1] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, IEEE 802.11 WG part a/b/g Std., 1999-2003.

[2] G. Holland, N. Vaidya, and P. Bahl, "A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks," in *ACM MobiCom*, July 2001.

[3] "ns-3 Project," http://www.nsnam.org/.

[4] "Madwifi: Multiband Atheros Driver for WiFi," http://sourceforge.net/projects/madwifi/.

[5] M. K. Simon, A. M. Hinedi, and W. C. Lindsey, "Digital Communication Techniques, Signal Design and Detection," 1995, printice Hall.

[6] J. G. Proakis, "Digital Communications," 1995, 4th edition, McGraw Hill.

[7] M. K. Simon and M. S. Alouini, "Digital Communication over Fading Channels," 2004, 2nd edition, John Wiley.

[8] A. systems, "WaveLAN 802.11b chipset for Standard Form Factors," December 2002, preliminary Product Brief.

[9] M. Lacage, M. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: A Practical Approach," INRIA No. 5208, Tech. Rep., May 2004.

[10] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *IEEE Infocom*, 2003.

[11] D. Qiao, S. Choi, A. Jain, and K. Shin, "MiSer: an optimal low-energy transmission strategy for IEEE 802.11a/h," in *ACM MobiCom*, 2003, pp. 161–175.

[12] V. Kanodia, A. Sabharwal, and E. Knightly, "MOAR: A Multi-channel Opportunistic Auto-rate Media Access Protocol for Ad Hoc Networks," in *IEEE Broadnets*, October 2004.

[13] A. Kamerman and L. Monteban, "WaveLAN II: A high-performance wireless LAN for the unlicensed band," Bell Labs Technical Journal, Tech. Rep., Summer 1997.

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

[15] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, "Opportunistic Media Access for Multirate Ad Hoc Networks," in *ACM MobiCom*, September 2002.

[16] J. d. Prado and S. Choi, "Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement," in *ICC*, May 2003.

[17] Z. Ji, Y. Yang, J. Zhou, M. Takai, and R. Bagrodia, "Exploiting Medium Access Diversity in Rate Adaptive Wireless LANs," in *ACM MobiCom*, September 2004.

[18] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *ACM MobiCom*, Cologne, Germany, August 2005.

[19] J. Bicket, "Bit-rate selection in wireless networks," Master's thesis, Massachusetts Institute of Technology, February 2005.

[20] I. Haratcherev, K. Langendoen, R. Lagendijk, and H. Sips, "Hybrid Rate Control for IEEE 802.11," in *MobiWAC*, 2004.

[21] I. Haratcherev, J. Taal, K. Langendoen, R. Lagendijk, and H. Sips, "Automatic IEEE Rate Control for Streaming Application," *Journal of Wireless Communication and Mobile Computing (JWCMC)*, vol. 5, no. 4, pp. 421–437, June 2005.

[22] Z. F. Li, A. Das, A. K. Gupta, and S. Nandi, "A Full Auto Rate (FAR) MAC Protocol for Wireless Ad Hoc Networks," *IEE Proceedings Communications*, to appear.

[23] D. Qiao and S. Choi, "Goodput Enhancement of IEEE 802.11a Wireless LAN via Link Adaptation," in *ICC*, June 2001.

[24] J. C. Wu, H. Liu, and Y. Lung, "An Adaptive Multirate IEEE 802.11 Wireless LAN," in *International Conference on Information Networking*, 2001, pp. 411–418.

[25] Q. Pang, S. Liew, and V. Leung, "A Rate Adaptation Algorithm for IEEE 802.11 WLANs Based on MAC-Layer Loss Differentiation," in *IEEE BroadNets*, October 2005.

[26] I. Tinnirello and S. Choi, "Temporal Fairness Provisioning in Multi-Rate Contention-Based 802.11e WLANs," in *IEEE WoWMoM*, June 2005, pp. 220–230.

[27] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," *ACM Communications*, vol. 19, no. 5, pp. 395–404, July 1976.

[28] C. Hoffmann, M. Manshaei, and T. Turletti, "CLARA: Closed-Loop Adaptive Rate Allocation for IEEE 802.11 Wireless-LANs," in *IEEE WirelessCom*, June 2005.

[29] M. Lacage, M. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: A Practical Approach," in *ACM MSWiM*, 2004.

[30] G. Bianchi, "Performance Anaylsis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 18, no. 3, March 2000.