School of Informatics, The University of Edinburgh

Institute for Computing Systems Architecture

Understanding the Role of Multi-Rate Retry Mechanism for Effective Rate Control in 802.11 Wireless LANs

by

Neda Koçi and Mahesh K. Marina

Informatics Research Report 1335

School of Informatics **July 2009**
http://www.informatics.ed.ac.uk/

Understanding the Role of Multi-Rate Retry Mechanism for Effective Rate Control in 802.11 Wireless LANs

Neda Koçi and Mahesh K. Marina

Informatics Research Report 1335

SCHOOL *of* INFORMATICS
Institute for Computing Systems Architecture

July 2009

A shorter version of this paper will appear in *Proc. 34th IEEE Conference on Local Computer Networks (LCN)*, Zurich, Switzerland, October 2009.

Abstract : We consider the multi-rate retry (MRR) capability provided by current 802.11 implementations and carry out simulation-based study of its impact on performance with state-of-the-art rate control mechanisms in typical indoor wireless LAN scenarios. We find that MRR is more effective in non-congested environments, necessitating the need for a mechanism to differentiate between congested and non-congested situations to better exploit the MRR capability. We also observe that decoupling the long-term rate adaptation algorithm from the MRR mechanism is key to fully realizing the benefits of MRR. Guided by these observations, we develop a new rate control mechanism called SDRA that performs coherent yet independent rate adaptation and multi-rate retry. SDRA's ability to identify the network situation and apply a suitable MRR strategy enables it to achieve similar or better performance compared to existing rate control mechanisms even with a simple rate adaptation algorithm. Separate rate adaptation and MRR in SDRA, both based on local SNR measurements, leads to its superior performance over existing mechanisms in congestion scenarios.

Keywords : 802.11 wireless networks, Rate adaptation algorithms, Multi-rate retry.

1 Introduction

In recent years, IEEE 802.11 wireless local area networks (WLANs) have become increasingly popular to become the dominant technology for indoor broadband wireless networking. The 802.11 (a/b/g) physical layer provides multi-rate capabilities. The overall mechanism responsible for selecting a transmission rate from among multiple available transmission rates at run time is referred to as the *rate control*. Rate control is recognized to play a critical role in determining the WLAN performance. This together with the fact that 802.11 MAC/PHY specifications leave the rate control mechanism to vendor discretion has led to many rate control proposals (e.g., [1, 2, 3, 4, 5]). Most of these proposals tend to focus on a strategy where multiple (re)transmits of a frame use the same rate even though current 802.11 implementations support the capability to change the rate between frame retransmits. This latter capability is the focus of this paper and henceforth referred to as *multi-rate retry (MRR)*.

The rate control mechanism in 802.11 can be seen to be composed of two interrelated mechanisms: (1) *rate adaptation algorithm (RAA)* responsible for determining the initial transmission rate of a frame and adaptation of that rate over time; (2) *MRR mechanism* responsible for selecting the transmission rate to be used for any frame retransmissions. The rate adaptation algorithm attempts to track the longer term fluctuations of the channel through link layer statistics or physical layer (PHY) based metrics, whereas the MRR mechanism is left to deal with short-term variations of the channel. Even though the MRR capability is implemented by the widely used MadWifi driver [6] and is part of all RAAs provided by the driver, its impact on rate adaptation performance surprisingly has not been studied in detail to date.

In this paper, we first conduct a detailed simulation study investigating the impact of MRR with state-of-the-art rate control mechanisms [2, 3, 4] in typical indoor wireless LAN scenarios. Our study shows that the specifics of RAAs and associated MRR mechanism expectedly plays a crucial role in determining the MRR impact. More importantly, we find that MRR is particularly beneficial in non-congested scenarios, thus motivating the need for a mechanism to distinguish between congested and non-congested situations to fully exploit MRR capability. We also observe that the coupling between RAA and MRR in existing rate control mechanisms limits the effectiveness of MRR. Based on these findings, we propose a new rate control mechanism (termed SDRA) that performs coherent yet independent coherent rate adaptation and multi-rate retry. The uniqueness of SDRA stems from the fact that it relies on local SNR measurements for rate adaptation as well as selecting appropriate MRR strategy depending on the network situation. Our evaluations not only demonstrate that SDRA meets its intended goals, but also show that it outperforms existing rate control mechanisms in scenarios with multiple contending stations due to its ability to separate RAA and MRR.

The remainder of this paper is structured as follows. Next section reviews the rate adaptation algorithms considered in this work and presents the state of the art on multi-rate retry mechanisms. Section 3 describes the simulation environment. In Section 4, we quantify the impact of MRR with existing rate control mechanisms in typical wireless LAN scenarios and draw a set of guidelines for effective rate control mechanisms that can exploit MRR capability. Based on these guidelines, Section 5 develops a new rate control mechanism called SDRA and presents simulation results to demonstrate its effectiveness. Finally, we conclude in Section 6.

2 Background and Related Work

2.1 Rate Adaptation Algorithms

A number of rate adaptation algorithms (RAAs) have been proposed in recent years. In this study, we consider three well-known RAAs: ARF [1] and its variant AMRR [2]; Onoe [3]; and SampleRate [4]. We review these algorithms below.

Auto Rate Fallback (ARF) [1] is the first rate adaptation algorithm proposed in the literature. Originally developed for Lucent Technologies, is the most widely implemented RAA due to its simplicity. It requires very little state to alternate the transmission rates, by keeping track of timing functions as well as missing ACK frames (signifying frame transmission failures). The sender adjusts the rate upwards after a fixed number of successful transmissions (10 consecutive ACKs), and adjusts the rate downwards after one or

two consecutive failures. If after an upwards rate adjustment the transmission fails, the rate is immediately readjusted downwards. ARF is conservative in reacting to fluctuations in the channel condition.

Adaptive Multi Rate Retry (AMRR) [2] is a two-stage rate adaptation technique proposed as an extended variant of ARF. The algorithm deals with short-term fluctuations in the channel conditions via MRR (discussed in the next subsection) while performing long-term adaptation by applying an ARF-like mechanism. Henceforth, we use the terms ARF and AMRR synonymously.

Onoe [3], the algorithm developed by creators of MadWifi, follows a credit-based strategy in that it maintains credits for the currently used rate on a per-destination basis to aid in the decision to increase the data rate. Initially, the rate is set to 24Mbps for 802.11a/g, and 11Mbps for 802.11b, with zero credits for that rate. Subsequently, Onoe decides on the rate and updates credits periodically after every observation interval (default one second). It steps down to a next lowest rate if either none of the transmissions were successful in the previous interval, or more than ten frames were transmitted with average retries exceeding one. Credit count is decremented if more than 10% of the frames are retried during the previous interval, and incremented otherwise. If the credit count reaches a threshold (10) then Onoe shifts to the next higher rate. Clearly, Onoe is conservative in moving to higher rates; it takes at least ten observation intervals for a rate increase, whereas rate decrease can happen in just one interval.

SampleRate [4], differently from previous algorithms, lays special emphasis on lossy links (typical in outdoor environment). The algorithm selects the data rate on a per-frame basis. Initially, it uses the highest possible rate (11Mbps in 802.11b and 54Mbps in 802.11a/g). Afterwards, the eligible rate with smallest estimated average frame transmission time (including any retransmissions for loss recovery) is selected. A data rate is eligible if successful transmission time without retry using that rate is smaller than the average transmission time with current rate and the use of that rate did not previously result in four consecutive transmission failures. To estimate the frame transmission times at eligible rates other than the current rate, every tenth frame is sent using a rate randomly chosen from the set of other eligible rates.

2.2 Multi-Rate Retry

Multi-Rate Retry (MRR) is a mechanism first introduced by the MadWifi driver project [6]. This project provides one of the most advanced Linux driver for 802.11a/b/g network interface cards (NICs) based on Atheros chipsets. The driver itself is open source but depends on the proprietary *Hardware Abstraction Layer (HAL)* that is available in binary form only. The driver maintains several FIFO (First In First Out) queues of transmission descriptors to schedule packets for transmission. Each transmission descriptor contains detailed control information related to a frame's transmission. The most important and relevant information contained by the descriptor is an ordered set of four pairs of rate and transmission count fields (r_0/c_0, r_1/c_1, r_2/c_2, r_3/c_3) that we refer to as the *multi-rate retry chain*. Whenever the wireless medium is available for transmission, the hardware triggers the transmission of the descriptor located at the head of the FIFO. Originally the frame is transmitted with the rate r_0 specified in the descriptor. If this transmission fails (determined by the absence of an ACK response), the hardware attempts to retransmit the frame with the rate r_0 up to c_0 - 1 times. If the transmission is still unsuccessful, 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 frame is discarded. Note that the sum of all counters implicitly indicate the frame retry limit.

If the MRR mechanism is disabled or a fixed rate is used for frame transmissions (the latter implying that the RAA is disabled[1]), the frame will be (re)transmitted in total for ten times by default in the MadWifi implementation. However, if MRR is enabled, then the number of transmission attempts for a frame is determined by the retry chain parameters corresponding to the RAA used.

The current stable release (v0.9.4) of the MadWifi driver provides four RAAs [2, 3, 4, 5] with SampleRate as the default algorithm[2]. MRR functionality is also enabled by default and is used by all the algorithms. The general understanding justifying the MRR usability is that RAAs are able to estimate the best long-term transmission rate to use and because of some short term fluctuation of the channel some frames are lost. To recover quickly from this state, some fast countermeasures should be undertaken such

[1] Disabling RAA implicitly means MRR is also disabled.

[2] Minstrel [5] is a variant of the SampleRate algorithm. It dynamically adjusts the multi-rate retry chain parameters in contrast to the static approach adopted by other rate control mechanisms. However, the specific adjustment mechanism chosen is tightly coupled to TCP traffic, so we do not consider this mechanism in order to make our study relevant for all types of applications.

RAA	Tx Rate	Count
AMRR [2]	r_0	1
	r_1= r_0 - 1	1
	r_2= r_0 - 2	1
	r_3= lowest rate	1
Onoe [3]	r_0	4
	r_1= r_0 - 1	2
	r_2= r_0 - 2	2
	r_3= lowest rate	2
SampleRate [4]	r_0	2
	r_1= ${r_0}^a$	3
	r_2= lowest rate	3
	r_3= 0	0

[a] If no ACKs are registered yet for the specific rate r_0, then r_1 is set to the lowest rate.

Table 1: Multi-rate retry chain parameters.

as lowering the rate for ensuing retransmissions of a frame without having to wait for the adjustment of the long-term rate. Table 1 summarizes how the parameters of the retry chain are initialized for each of the RAAs considered in this work.

As specified by the 802.11 MAC protocol, retransmission of a frame is initiated when the timer to receive an ACK (in response to a previous frame transmission attempt) expires. The absence of an ACK implies that the previous transmission attempt failed (either due to frame corruption by the channel or due to collision with another frame). As can be seen from Table 1, existing MRR mechanisms retry several times using the current rate, implicitly assuming that either a collision has occurred or that the channel condition has temporarily changed. If the retransmission is still unsuccessful, then lower rates are used. Using an extensive number of retries at the current rate may increase the probability of a successful transmission, but it can also increase the overall frame transmission delay, thus lowering throughput.

We conclude this section by noting that MRR mechanisms outlined above have similarity to the incremental redundancy and hybrid ARQ approaches explored in the context of cellular networks. Note that in contrast to MRR, incremental redundancy is applied at the physical layer with greater receiver feedback. Recently, mechanisms such as PPR [7] aim to improve the utilization of available capacity by selectively retransmitting only those bits that get corrupted over the air. However, neither traditional incremental redundancy mechanisms nor PPR-like mechanisms are directly implementable with commodity 802.11 hardware as they require more flexible and configurable radios such as software-defined radios.

3 Simulation Environment

Our evaluations are based on simulations using the well-known QualNet simulator (version 4.0) [8]. We implemented SampleRate, Onoe and MRR functionality in QualNet, strictly following the MadWifi implementation.

Table 2 lists the values for simulation parameters common to all our experiments; these parameters are compliant with the IEEE 802.11a standard considered in our evaluations.

The frame retry limit is a key MAC layer parameter in our work. The 802.11 standard defines a long and a short retry limit with seven and four as the suggested values for these two limits, respectively. However, the default retry limits of some wireless cards differ from the standard. MadWifi implementation of the 802.11 protocol uses a default value of ten for the retry limit when MRR is disabled[3]. When the MRR mechanism is enabled, the maximum number of transmission attempts for a frame is based on the retry

[3] Note that the MadWifi implementation lets the user customize this value.

Physical Layer		MAC Layer	
Frequency band	5 GHz	MAC Protocol	DCF
Noise floor	-94 dBm[a]	CWmin	15
Tx Power	17 dBm	CWmax	1023
Max. Rx Sensitivity	-65 dBm	RTSThresh	2043 Byte
Antenna	Omni	SIFS time	16μs
Antenna Efficiency	0.8	SLOT time	9μs
Antenna Loss	0.5 dB	Access Mode	Basic
Antenna Height	1.5m	Connection	AP Mode

[a]The background noise level is set to -94 dBm as suggested by measurements presented in specialized studies [9].

Table 2: Simulation Parameters.

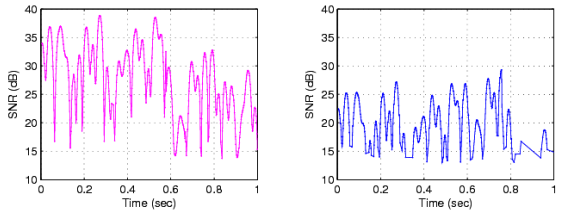


Figure 1: SNR vs. time trace for two sample indoor wireless links.

chain parameters of the RAA, i.e., $c_0 + c_1 + c_2 + c_3$. For example, SampleRate's MRR mechanism uses a retry limit of eight as shown in Table 1.

Channel Model. We consider indoor 802.11 scenarios. To model the radio propagation in indoor environments, we use a channel model that is composed of a large-scale path loss model, a log-normal shadowing model and a Ricean fading model. The average path loss PL (in dB) for a transmitter and receiver separated by distance d is calculated as:

$$PL_{dB} = PL_{dB}(d_o) + 10\alpha log(\frac{d}{d_o}) + X_\sigma + \textit{Fading} \quad (1)$$

where X_σ is a zero-mean Gaussian distributed random variable with standard deviation σ and α denotes the path loss exponent. We set α to 3.0 and σ to 4.0, which are reasonable values for the indoor environment. To account for environment mobility, we set the maximum velocity of objects in the environment to 1 m/s. Fig. 1 shows the Signal-to-Noise Ratio (SNR) over time observed at the receiver for two sample indoor wireless channels simulated in our study.

4 MRR Impact on Performance

In this section, we study the impact of multi-rate retry (MRR) mechanism on 802.11 rate control performance. The key metric of interest is the application-level throughput computed as the amount of application data delivered at the destination per unit of time. We refer to this metric as *goodput* in the paper and measure it in Mbps. Maximizing goodput is the goal of all rate control techniques. We also consider two additional metrics: (a) *loss ratio* — the ratio of number of frames discarded by the MAC layer (after failing to successfully transmit even after several retries up to the specified frame retry limit) to the total number of original frames transmitted; (b) average number of retransmissions per frame (in short, *retx ratio*). Clearly, smaller loss ratio and smaller retx ratio are preferred.

We consider two scenarios:

- Scenario I: This is a simple scenario that models communication between an Access Point (AP) and a client station separated by a distance d; we present results obtained from averaging over a large number of randomly chosen d values. This scenario is representative of a home WLAN setting.
- Scenario II: This scenario reflects a typical office WLAN scenario where multiple client stations contend to communicate with a AP node. Specifically, we consider four client stations distributed around the AP for our experiments. Note that collisions can occur in this scenario due to simultaneous transmission by multiple stations as well as due to hidden terminals (two stations hidden from each other communicating with an intermediate AP node).

In both scenarios, we setup a CBR/UDP traffic flow between each client station and the AP. We use fixed 1KB frame size. We consider a worst case situation where each flow is continuously backlogged, i.e., each flow generates enough traffic to saturate even a link with maximum capacity (54Mbps). Each data point presented is the result of averaging over multiple simulation runs, each having a duration of 180s.

4.1 Scenario I Results

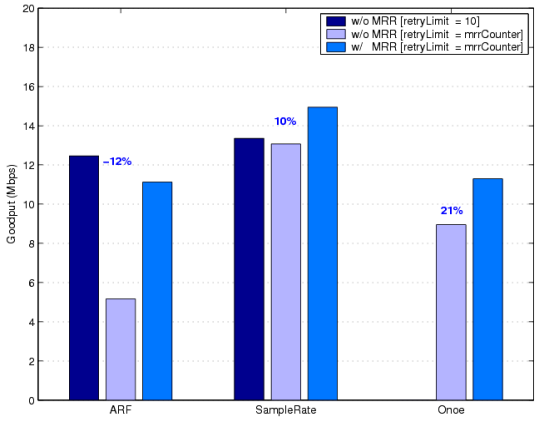


Figure 2: Impact of MRR on goodput performance with ARF, SampleRate and Onoe in Scenario I.

	ARF[a]		SampleRate		Onoe	
Loss Ratio	0.2	0.4	0.1	0.09	0.1	0.05
Retx Ratio	5.7	3.6	4.3	4.3	5.0	4.1

[a]the right-hand column for each algorithm denotes the values obtained when MRR is enabled.

Table 3: Impact of MRR on loss ratio and retx ratio with ARF, SampleRate and Onoe in Scenario I.

Fig. 2 shows the impact of MRR with different RAAs in this scenario. To have a fair evaluation and to clearly identify the impact of MRR, we compare the goodput for each RAA in three cases: (1) with MRR disabled and retry limit set to ten, the default value in the MadWifi implementation; (2) with MRR disabled but retry limit set to the total number of retries specified by the MRR scheme corresponding to each RAA (i.e., four for ARF, eight for SampleRate and ten for Onoe); and (3) with MRR enabled and retry limit set to the total number of retries specified by the MRR scheme corresponding to each RAA. Goodput performance with ARF and SampleRate is represented by three bars following the above specified order, while Onoe performance is represented by two bars with and without MRR mechanism enabled — dark blue bar and light grey bar for Onoe are identical.

From Fig. 2, the main observation is that the impact of MRR is different with different RAAs. However, this is expected given that each RAA and the corresponding MRR retry chain parameters are different. Comparing the two blue bars for ARF and SampleRate and the two bars from Onoe, we also observe that MRR reduces goodput with ARF/AMRR, whereas it the other way around with SampleRate and Onoe — the percentage values in the figure indicate the amount of goodput reduction/improvement. To better understand the differences in MRR impact with different RAAs, Table 3 shows the results for loss ratio and retx ratio. To ease the analysis, results are shown for only two cases for these two metrics — retryLimit=10 when MRR is disabled and retryLimit set based on MRR retry chain parameters of the RAA when MRR is enabled.

Looking at ARF results, we see that retx ratio is reduced with MRR, whereas loss ratio increases. This is because the MRR mechanism associated with ARF permits only a small number of retries (specifically, four). While fewer retries aid in improving the responsiveness of the RAA, it can also increase losses as confirmed by the results. Greater number of frame losses could lead to disastrous performance of applications sensitive to loss. For example, if the application is based on TCP, goodput will be further reduced; for each frame loss reported by the MAC layer to higher layers, TCP will initiate its own loss recovery procedure, increasing the time to successfully deliver a packet and lowering the goodput. SampleRate, on the other hand, benefits (albeit marginally) from using MRR. This is because MRR helps in recovering from frame transmission failures arising from aggressive use of higher bit-rates, which is confirmed by the slight reduction in losses with MRR. Onoe benefits the most from using MRR as seen from the substantial improvement across all metrics. This is because Onoe multi-rate retry chain provides enough opportunity to recover from frame transmission failures arising due to short-term channel fluctuations, which in turn translates into improvement in goodput. However, the extent of goodput improvement is limited by the conservative nature of the Onoe RAA. In this scenario, SampleRate performs better overall with and without MRR.

4.2 Scenario II Results

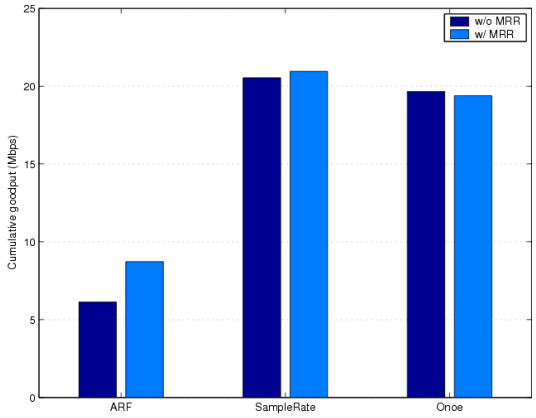


Figure 3: Impact of MRR on goodput performance with ARF, SampleRate and Onoe in Scenario II.

	ARF		SampleRate		Onoe	
Loss Ratio	0.075	0.425	0.013	0.025	0.013	0.008
Retx Ratio	5.7	4.1	4.6	4.8	4.6	4.6

Table 4: Impact of MRR on loss ratio and retx ratio with ARF, SampleRate and Onoe in Scenario II.

We now look at the impact of MRR in the presence of channel contention using scenario II described earlier. In a congested environment, the rate control problem is more difficult because RAAs have to

additionally guard against misinterpreting frame transmission failure due to multiple access collision as channel induced errors and vice-versa. Recent work [10] presents comparative analysis of the performance of RAAs in a congested scenario, showing that RAAs that rely purely on link-layer information (e.g., frame transmission failures, transmission time) perform poorly, especially with increasing number of active nodes in the network.

Fig. 3 and Table 4 show the impact of MRR on performance with each of the RAAs in scenario II. Note that here we only consider two cases for each RAA: (1) with MRR disabled and retry limit set to ten, the default value in the MadWifi implementation; and (2) with MRR enabled and retry limit set to the total number of retries specified by the MRR scheme corresponding to each RAA (i.e., four for ARF, eight for SampleRate and ten for Onoe).

These results show that, in contrast to the previous scenario, ARF's goodput improves with MRR in presence of collisions. This is because it benefits from fewer number of retries which is more effective in congested scenarios. However, loss ratio also increases as before. Getting a higher goodput at the expense of link reliability may not be a desirable thing from the application standpoint as it may hurt overall quality of service for some applications. We also observe that ARF performance degrades with increasing number of active nodes in the network (results not shown for brevity). SampleRate goodput is better compared to other algorithms in presence of contention. Though loss ratio with SampleRate is doubled from using MRR, it is still quite small to have any undesirable impact on application performance. Onoe with MRR enabled shows fewer losses, but also a slight decrease in goodput. Nevertheless, its performance is still satisfactory to make it a suitable choice for applications demanding high reliability.

4.3 Summary

Comparing results from the two scenarios, we observe that MRR is more effective in a non-congested environment, thus necessitating a mechanism to differentiate between congested and non-congested situations in order to better exploit MRR capability. Results also show that the specific characteristics of the RAA and multi-rate retry chain parameters play a big part in determining the performance impact of MRR. Through our implementation and evaluation, we have also observed that the statistics used by RAAs become unreliable due to coupling between RAA and MRR, thus reducing the net benefit from using MRR.

5 SNR-based Differentiated Retry Algorithm

In light of the observations made in the previous section, we propose a new rate control mechanism in this section termed SDRA that performs coherent rate adaptation and multi-rate retry. In particular, SDRA uses SNR (a PHY layer metric) to directly estimate the channel quality to avoid undesirable coupling between the RAA and the MRR mechanism. SDRA additionally uses SNR measurements to differentiate between congested and non-congested situations, and employs a different multi-rate retry chain in each of those situations.

5.1 Design

5.1.1 Rate Adaptation Algorithm

The algorithm to adapt the long-term frame transmission rate in SDRA is simple and based on local SNR measurement, which can be easily obtained from the driver and NIC firmware upon a frame reception. Our rate adaptation algorithm consists of two sub-mechanisms: (i) *SNR estimation*; (ii) *rate selection*. Both of these are done on a per-destination basis.

SNR estimation mechanism is inspired from recent work by Judd et al. [11]. After receiving a frame at the physical layer, received SNR value is reported to the MAC layer. Based on link symmetry and reciprocity theorem[4], we can suppose that average SNR value measured at a sender node using frames from a receiver node is same as that at the receiver node. So even though the sender does not know the

[4]Reciprocity theory states: "If the role of the transmitter and receiver are instantaneously interchanged, the signal transfer function between them it remains interchanged".

instantaneous channel quality at the receiver before selecting the transmission rate for a frame destined to the receiver, it can assume that the SNR value reported by the PHY for the previous frames from the receiver reflect the required channel quality at the receiver. To estimate the long-term SNR_{Avg} value that describes the channel quality for rate selection, we use instantaneous values reported by the PHY to do time-based averaging. Specifically, every time a fresh SNR value (referred to as SNR_{Cur}) is reported by the PHY for a destination, we update the estimate of SNR_{Avg} for that destination as follows:

$$SNR_{Avg} = \frac{SNR_{Avg} * f(\Delta t) + SNR_{Cur}}{1 + f(\Delta t)} \quad (2)$$

where f(Δt) is a linearly decreasing function starting at 1 and decreasing to 0 when Δt exceeds a decision time window. Since the indoor wireless channels can exhibit short transient fades that last for as little as the duration of a single frame transmission, to avoid high fluctuation of the long-term predicted value we detect and filter out transient fades by only considering those SNR_{Cur} values in the updating process that have a difference from SNR_{Avg} larger than a predefined deviation threshold for at least two consecutive packets. So if we first detect a SNR sample that deviates from the average by more than the threshold, the sample is marked as transient fading and the SNR updating procedure using that sample is postponed and eventually carried out only if subsequent samples also fall under the same category. In our implementation, we set the decision time window to two seconds, and the SNR deviation threshold to 7dB.

A straightforward approach for rate selection to a destination is to use the estimated channel quality (i.e., SNR_{Avg}) for the destination to lookup the highest rate which ensures that the bit/frame error rate is under a desired threshold. However, this approach can be conservative as using a somewhat higher rate can still result in successful transmission with high probability. Thus we select a rate that is higher[5] than that suggested by the straightforward approach.

5.1.2 Retry Limit and Multi-Rate Retry Chain

Judiciously choosing the retry limit is crucial because it affects loss probability and frame transmission time which in turn determine the overall efficiency (goodput) and reliability. If it is set to a small value, it will improve the responsiveness of the RAA to channel fluctuations, but also increases loss probability as seen in Section 4. A large retry limit, on the other hand, can increase the average frame transmission time that in turn can reduce the goodput. Using a large retry limit does not necessarily imply that maximum number of retries is always used. In fact, if the multi-rate retry chain parameters are appropriately chosen, the frame transmission can still succeed with a few retries. So to retain the flexibility for exploiting the MRR capability, we set the retry limit to ten (just as in Onoe and the default limit used in MadWifi in the absence of MRR).

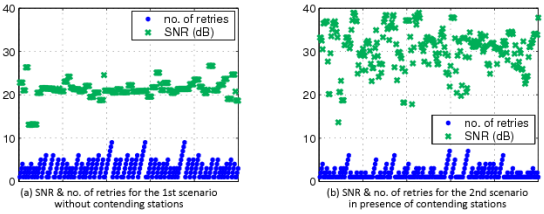
In deciding the rate/count (r_i/c_i) pairs for the multi-rate retry chain in SDRA, we distinguish between two situations and use distinct retry chain parameters for each of those situations, as described below:

1. With short-term channel fluctuations alone, the MRR mechanism needs to be responsive. In such a situation, we propose to carefully increase the introduced redundancy by selecting lower retransmission bit-rates after each transmission attempt failure (determined at the sender by the absence of an ACK response from the receiver). Specifically, if the frame transmission using the long-term bit-rate (provided by the rate adaptation algorithm) is not successful then we propose to retransmit the frame with the same rate for the next retry and, if necessary, switch to the next lower rate for the next two retries. If frame transmission is still not successful then we propose retransmitting the frame using the next lower rate for three more times and then drop to the lowest transmission bit-rate for the remaining retries. Thus, retry count setting in the SDRA multi-rate retry chain is as follows: $c_0 = 2$, $c_1 = 2$, $c_2 = 3$, $c_3 = 3$. This setting differs from previous proposals which are either too aggressive or too conservative in selecting lower bit-rates for retransmission.

2. In presence of collisions or fast fading, the above approach is not appropriate as selecting lower bit-rates for retransmission when higher bit-rates can be used leads to lower throughput compared to

[5]Rate that is two levels higher than the one estimated by the straightforward approach is used in our implementation. This is of course feasible only if the estimated rate is lower than 48Mbps, otherwise 54Mbps is selected as the rate.

Figure 4: SNR-Retry relation with/without contending stations.



the case when MRR is disabled. In this situation, our approach is completely different from the one proposed above. Specifically, we propose the use of the current rate for the next four retransmissions. If the frame transmission is still unsuccessful after five transmission attempts, we retry the transmission using the next lower rate two times and then further increase the redundancy by lowering the rate by another level for subsequent two retries. If still unsuccessful, the final retransmission is made using the lowest available rate. This results in the following retry count setting for the multi-rate retry chain: $c_0 = 5$, $c_1 = 2$, $c_2 = 2$, $c_3 = 1$. Indeed, the approach just described is the opposite to the previous one in that we increase the redundancy only when the failures are persistent at the current rate.

5.1.3 Loss Differentiation

In order to employ distinct multi-rate retry chains in different situations as described above, we need a mechanism to identify the cause behind a frame transmission failure. This problem, commonly referred to as the loss differentiation problem, has received attention from the research community recently [12, 13, 14]; it is still an active research topic with no single loss differentiation solution available for all situations. In our MRR context, we need a way to infer the cause behind the failure of first transmission attempt of a frame so that an appropriate retry chain among the two options can be selected for efficient failure recovery.

A natural approach to differentiate between different types of transmission failures is to use the SNR measurements from recently received frames. If these measured values are relatively high yet the transmission attempts are failing, then this could be because of collisions or transient fades. To further investigate this hypothesis, we conducted an experiment for both situations (i.e., with/without contending stations) and collect the last reported SNR measurements prior to each frame (re)transmission. Fig. 4 shows the relationship between measured SNR values and number of retries for both situations. From this figure, it can be clearly seen that in presence of contending stations, transmission of a frame can fail even for high values of SNR (30–40dB); this is not true when there are no contending stations. Thus this experiment validates our hypothesis that measured SNR values can aid in inferring the cause of a frame transmission failure.

Therefore, in SDRA, if SNR_{Cur} is greater than 20dB and frame transmission fails, then we conclude that collision has occurred or channel is exhibiting transient fades. When this happens, we use the *second* retry chain for failure recovery. The *first* retry chain is used as the default, but is overridden with second retry chain in presence of collisions and fast fading.

5.2 Evaluation

Here we evaluate the effectiveness of the proposed SDRA rate control mechanism in the same two scenarios used in Section 4.

Table 5 reports performance results with SDRA in scenario I. RAA in SDRA uses SNR estimation to infer the long-term channel condition and is expected to have a stable behavior even with short-term channel fluctuations. When MRR is enabled the achieved goodput with SDRA is increased by 27%. This improvement is due to two important features of SDRA: (i) since additional redundancy is gradually introduced with each additional retry, the transmission failures due to short-term channel fluctuates are dealt with efficiently; (ii) MRR mechanism in SDRA also helps in recovering from failures caused due to mis-

	MRR disabled	MRR enabled
Goodput (Mbps)	10.1	13.8
Loss Ratio	0.23	0.11
Retx Ratio	6.0	12.0

Table 5: SDRA performance in scenario I.

	MRR disabled	MRR enabled
Goodput (Mbps)	24.6	24.4
Loss Ratio	0.0125	0.01
Retx Ratio	4.43	4.48

Table 6: SDRA performance in scenario II.

estimation of long-term rate (specifically, choosing a higher rate than what is required to ensure reliable delivery) by the SNR-based RAA mechanism. An interesting observation from this experiment is that the performance with the combination of a simple rate adaptation algorithm and an optimized MRR mechanism in SDRA is comparable to sophisticated and complex RAAs seen earlier. For example, SDRA with MRR enabled outperforms Onoe.

Table 6 reports performance results with SDRA in scenario II. A key observation from these results is that SDRA delivers greater goodput in this scenario than all existing mechanisms (with and without MRR enabled) considered in this work. This is because of the decoupling of RAA and MRR mechanism in SDRA by directly estimating the channel quality using SNR measurements. In contrast, other mechanisms will adjust the rate based on frame failure and transmission time estimates measured in presence of collisions. Lastly, SDRA performance with MRR enabled is similar to SDRA without MRR, which is what is desired because MRR has limited value in scenarios with collisions (as observed previously in Section 4).

6 Conclusions

In this paper, we have focused on the multi-rate retry capability available with current 802.11 implementations and studied its impact on performance with state-of-the-art rate control mechanisms in typical indoor wireless LAN scenarios using simulations. Our study has shown that MRR is more effective in non-congested environments, thus it is important to identify such situations to fully exploit the MRR capability. We also observe that the potential benefit from MRR with existing rate control mechanisms is limited due to the coupling between RAA and MRR in those mechanisms. These crucial insights guide our design of a new rate control mechanism called SDRA that is based on SNR and applies a different MRR strategy depending on the presence of congestion. Simulation-based evaluation of SDRA shows that its intelligent selection of appropriate strategy when combined with a simple PHY-based rate adaptation algorithm can achieve similar or better performance than complex existing mechanisms. We also find that SDRA's ability to decouple long-term rate adaptation from MRR leads to its superior performance over existing mechanisms in congestion scenarios.

In future, we plan to validate our simulation results on a indoor wireless network testbed. We also plan to investigate the potential benefits of MRR in more dynamic and outdoor 802.11 wireless network scenarios.

References

[1] A. Kamerman and L. Monteban. WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed Band. *Bell Labs Technical Journal*, 1997.

[2] M. Lacage, M. H. Manshaei, and T. Turletti. IEEE 802.11 Rate Adaptation: A Practical Approach. In *Proc. ACM MSWiM*, 2004.

[3] Onoe Rate Adaptation Algorithm. `http://madwifi-project.org/browser/madwifi/trunk/ath_rate/onoe`.

[4] J. Bicket. Bit-rate Selection in Wireless Networks. Master's thesis, MIT, Feb 2005.

[5] Minstrel Rate Adaptation Algorithm. `http://madwifi-project.org/browser/madwifi/trunk/ath_rate/minstrel`.

[6] MadWifi: Multiband Atheros Driver for WiFi. http://madwifi-project.org/.

[7] K. Jamieson and H. Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. In *Proc. ACM Sigcomm*, 2007.

[8] QualNet Network Simulator. http://www.scalable-networks.com.

[9] B. Fu et al. Wireless Background Noise in the Wi-Fi Spectrum. In *Proc. 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, 2008.

[10] K. Ramachandran et al. Scalability Analysis of Rate Adaptation Techniques in Congested IEEE 802.11 Networks: An ORBIT Testbed Comparative Study. In *Proc. IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM)*, 2007.

[11] G. Judd, X. Wang, and P. Steenkiste. Efficient Channel-aware Rate Adaptation in Dynamic Environments. In *Proc. MobiSys*, 2008.

[12] Q. Pang, S. Liew, and V. Leung. Design of an Effective Loss-Distinguishable MAC Protocol for 802.11 WLAN. *IEEE Communications Letters*, 9(9):781–783, Sep 2005.

[13] J. Kim, S. Kim, S. Choi, and D. Qiao. CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs. In *Proc. IEEE Infocom*, 2006.

[14] S. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust Rate Adaptation for 802.11 Wireless Networks. In *Proc. ACM MobiCom*, 2006.