

Efficient Collision Detection for Auto Rate Fallback Algorithm

Federico Maguolo*

University of Padova – Department of Information Engineering
Via Gradenigo 6/B, 35131 Padova, Italy
maguolof@dei.unipd.it

Mathieu Lacage, Thierry Turletti

Institut National de Recherche en Informatique et en Automatique (INRIA), Planète Project,
2004 route des Lucioles, 06902 Sophia Antipolis, France
{lacage, turletti}@sophia.inria.fr

Abstract

The physical rate adaptation in 802.11 is a deeply investigated, though still open issue. Since 802.11 uses the random access Distributed Coordination Function (DCF) mechanism to access the medium, collisions can occur when two or more stations want to transmit data simultaneously. The challenge of rate adaptation schemes is to adapt the physical transmission rate based on channel-related losses, i.e. collisions should not influence the choice of the rate. In this paper we propose a new rate adaptation algorithm that behaves like Auto Rate Fallback (ARF), but makes use of the RTS/CTS handshake, when necessary, to decide whether the physical transmission rate should be changed. Main advantages of this algorithm are its simple implementation and the good performance it attains in presence of collisions. We evaluate the performance of this new rate adaptation algorithm, comparing it with other well known algorithms, by using the new NS-3 simulator.

1 Introduction

The standard de-facto for indoor broadband wireless networking is the IEEE 802.11 [1]. This standard provides specification for the MAC (Medium Access Control) layer and for the PHY (physical) layer. This technology works in non-static environments, hence the standard provides a set of mechanisms to adapt the transmission to the system variations. Particularly, the PHY allows a set of transmission modes that can be used to react to the channel variations. Each PHY mode uses a specific modulation and channel coding scheme, offering different performance in terms of throughput and robustness against reception noise and interference. The IEEE 802.11a standard [2], which is considered in this paper, specifies eight PHY modes.

*This work has been done during the author's internship at the Planète project, INRIA.

The goal of a rate adaptation scheme is to select the PHY mode in order to maximize a given metric, which is typically the system throughput.

The mandatory medium access control mechanism specified in the IEEE 802.11 standard is called Distributed Coordination Function (DCF). This mechanism is based on the CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) random access scheme. Consequently, the transmission failures experienced by a station are due to both channel related noise and collisions. Moreover, most of network cards export a measure of link quality and RSSI (Receive Signal Strength Indicator), but their values are unreliable since the standard [1] does not specify how to measure them [3]. The widely-adopted ARF (Auto Rate Fallback) scheme [4] does not work properly in multi-user environments because it selects the transmission rate based on a certain number of consecutive unsuccessful or successful transmission attempts.

Some recently proposed rate adaptation schemes [5, 6] deal with the problem of packet loss due to contention. These algorithms face the contention issue with the RTS/CTS mechanism of IEEE 802.11 that will be describe later. The RTS/CTS handshake increases the data overhead, but it can decrease the collision duration in presence of long packets. Indeed, it is better to use this mechanism only when losses are due to collisions. So, algorithms like CARA [5] and RRAA [6] propose to switch on and off the mechanism based on some heuristics.

All the rate adaptation algorithms proposed in literature were evaluated in saturation condition, i.e. when nodes' queue never empties. However, when considering multimedia traffic, like Voice over IP (VoIP) or video streaming, this hypothesis is not satisfied. Infact, the time between two consecutive transmission attempts could be greater than the channel coherence time, so that the PHY rate used for the former transmission could be not anymore optimal. In this work, we propose a new rate adaptation algorithm, named AARF-CD (Adaptive Auto Rate Fallback with Collision Detection). It is a modification of

the Adaptive Auto Rate Fallback (AARF) scheme [7], where the RTS/CTS mechanism is turned on before deciding whether to decrease the rate while it is turned off after a variable number of successful transmission attempts. Our algorithm is easy to implement and we show that it achieves better performance than RRAA and CARA, in terms of system throughput. AARF-CD inherits the ARF timer which increases the PHY rate if no new PHY rates are tried for a certain period (time based timer). This feature could improve the performance when multimedia sources are used. However, for fair comparison, we consider the saturated traffic scenario that has been used in the literature to test the performance of the other algorithms.

Hence we compare our proposed algorithm with Auto Rate Fallback (ARF) [4], AARF, RRAA, CARA and with an ideal rate adaptation algorithm that always selects the best rate, according to the actual channel conditions. Performance evaluation are performed by using the new NS-3 simulator [8].

The remainder of the paper is organized as follows. Section 2 presents related works. Section 3 gives the system overview and describes the AARF-CD algorithm. Simulation results and comparisons with other schemes are shown in Section 4. Finally, Section 5 draws the conclusions.

2 Related works

Physical rate adaptation in IEEE 802.11 is a well known and deeply studied issue. Algorithms have been proposed in the literature and part of them cannot be implemented in the real network interfaces because they are not standard compliant. In this section, we describe the most known rate adaptation algorithms, bringing more details to the ones we compare with our algorithm.

ARF [4] is a widely adopted and well known rate adaptation algorithm. The decision whether to increase or decrease the transmission rate is based on the number of consecutive successfully or unsuccessfully transmission attempts, respectively. This algorithm is widely adopted because it is simple. The main problem of this algorithm is that it cannot distinguish between losses due to collision from losses due to channel, so it achieves poor performance in multi-user scenarios. Another problem, pointed out in [7], is that it tries a higher rate everytime it obtains a fixed number of successfully transmission attempts, even if the current rate is the most convenient. To alleviate this problem, the authors of [7] proposed the Adaptive ARF (AARF) algorithm that behaves like ARF with the difference that the number of consecutive successfully transmission attempts before trying the higher rate is incremented exponentially every time the higher rate transmission fails. AARF performs better than ARF in case of single-user scenarios, but it has the same problems as ARF in a multi-user scenarios.

In the Receiver Based Auto Rate (RBAR) scheme [9], the RTS/CTS handshake is mandatory. The receiver selects the best transmission rate on the basis of the RSSI measured during the reception of the RTS frame. The selected PHY mode is

communicated to the sender using a modified CTS frame, and the data frame is sent accordingly. This scheme is not standard compliant since it requires modifications to the RTS, CTS and data frames structure. It should be noted that this algorithm behaves like the ideal protocol described in the previous section, provided that the RSSI is accurate and the RTS and CTS frames are always sent using the lowest rate. The difference between the ideal algorithm and RBAR is that the latter has to use RTS/CTS, while it is optional in the former.

The Robust Rate Adaptation Algorithm [6] is composed by two different mechanisms: the rate selector, which selects the transmission rate based on the computation of the packet loss ratio observed in a finite window, and the Adaptive RTS, which turns on and off the RTS/CTS mechanism based on the results of the last transmission. The parameters of the rate selector depend on the used rate and the authors proposed default values for them. We will show in our simulations that, with these settings, RRAA does not always make the best choice for the rate (although the authors showed that it works better than ARF and AARF in a testbed).

The Collision-Aware Rate Adaptation (CARA) scheme proposed in [5] exploits the RTS/CTS mechanism for loss differentiation. CARA-RTS implements an adaptive RTS/CTS probing scheme that reduces the overall RTS/CTS usage. Moreover, the authors proposed CARA-BASIC, which is CARA-RTS with a further mechanism to detect collisions based on the Clear Channel Assessment (CCA) module. They show that their algorithm outperforms ARF in a multi-user scenario, but it is not compared with other mechanisms that are known to perform better, like RBAR.

3 System Overview

We consider a standard IEEE 802.11a WLAN with multiple users contending for the medium. We assume that all the stations are within carrier sense range, so that the hidden/exposed terminal issue is not present. The available rates using IEEE 802.11a are 6, 9, 12, 18, 24, 36, 48 and 54 Mbps, but we discarded the 9 Mbps mode from the set of selectable modes because the corresponding available throughput is less than the one of the 12 Mbps mode for any Signal to Noise Ratio (SNR) value [10].

We assume the reader is familiar with the IEEE 802.11 mechanism. On the contrary case we refer the reader to [10].

There are two ways to transmit a frame: basic access and RTS/CTS handshake. With the basic access, the frame is transmitted and if it is correctly received, the receiver sends back an acknowledge frame (ACK). With RTS/CTS, the packet transmission is preceded by a Request To Send (RTS) frame and the answer from the receiver with a Clear To Send (CTS) frame. The RTS and CTS frames are shorter than data frames, hence collisions among RTS frames are shorter than collisions among data packets.

We assume that the RTS, CTS and ACK frames are always

sent at the minimum rate (i.e. 6 Mbps), hence they are robust against channel-related losses. Moreover, we notice that, when the RTS/CTS mechanism is used in an infrastructure scenario, collisions can occur only during the transmission of the RTS frame. Therefore, the RTS/CTS mechanism is a good way to distinguish between collision errors and channel errors.

In the following, we describe more accurately how the existing rate adaptation algorithms work, and then we give an inside look at the proposed rate adaptation algorithm AARF-CD.

3.1 ARF, AARF

ARF is an ACK-based rate selection scheme proposed in [4]. The transmission rate is decreased when two consecutive transmission failures happen, and it is increased when ten consecutive transmission attempts succeed. If the rate is increased and the next transmission attempt fails, the rate will be immediately decreased. An additional timer is used to increase the rate when the same rate is used for a long period. This timer could be time-based or packet-based. This algorithm performs well when there are no contenders in the medium. However, it suffers in multi-user scenarios because it cannot distinguish losses due to channel errors from those due to contentions.

Adaptive ARF (AARF) is proposed in [7] to alleviate the inefficiency of ARF due to the automatic attempt of new rates every ten successful transmissions. This algorithm behaves like ARF with one difference: instead of trying the next higher rate every ten successful transmissions, it doubles this number whenever the first transmission attempt with the higher rate fails. The number of successful transmission attempts required to increase the rate is reset to ten every time the rate is decreased and it does not exceed a given threshold (which is equal to 50).

3.2 RRAA

The Robust Rate Adaptation Algorithm [6] includes two mechanisms: the rate selector, called RRAA-BASIC, and the Adaptive RTS (ARts). The rate selector counts the number of transmission failures that occur during an observation window. Then, at the end of this observation window, it computes the packet loss ratio and makes its decision: if the packet loss ratio is greater than a threshold called P_{MTL} (Maximum Tolerable Loss threshold), the rate is decreased; if the packet loss ratio is less than a threshold called P_{ORI} (Opportunistic Rate Increase threshold) the rate is increased, otherwise the current rate is maintained for the next observation window. The observation window length (which is in number of transmission units) and the thresholds values depend on the current rate. Moreover, the authors propose a mechanism to decrease the rate even if the observing window is not ended: every time a transmission failure is detected, the packet loss is computed assuming that the remaining transmission attempts in the observation window will succeed. If this packet loss estimation is greater than

P_{MTL} , the rate is immediately decreased (without waiting for the observation window termination). The ARts mechanism works simultaneously with the rate selector and it is independent from it. A counter is used to estimate how many transmission attempts can be made using the RTS/CTS mechanism. When the counter is greater than 0, the RTS/CTS mechanism is used and the counter is decreased by one for each attempt. It exists a RTS window, initially equal to 0, which is incremented by one when the RTS is not used and the last transmission attempt fails. It is halved when the RTS is used and the last transmission succeeds, or when the RTS is not used and the last transmission fails. Every time the RTS window value is modified, the counter is set to the window value. RRAA has been implemented in a testbed and it performs better than ARF and AARF when a multi-user scenario is considered, with or without hidden nodes [6].

3.3 CARA

The main mechanism of the rate selection in CARA is similar to ARF. There are two counters: the first one stands for the number of consecutive successful transmissions, while the second one counts the number of consecutive failures. When a packet arrives at the MAC, if its size is greater or equal than a given threshold, or if the number of consecutive transmission failures is greater than another given threshold (which is 1 by default), the RTS/CTS mechanism will be used. If the RTS/CTS handshake fails, the counters will not be modified; instead, if the RTS/CTS handshake succeeds, the data frame will be transmitted with the current rate, like when the RTS/CTS mechanism is not used. If the data transmission succeeds, the counter of successfully data transmissions will be incremented by 1, while the counter of consecutive data failures will be reset. If the number of consecutive successfully data transmission attempts reaches a given fixed threshold (which is by default equal to 10 like ARF), the rate will be increased. If the data transmission fails, the counter of consecutive data transmission failures will be incremented by one, while the counter of successfully data transmission attempt will be reset. If the counter of consecutive transmission failures reaches a threshold (which is equal to 2 by default), the current rate will be decreased. Notice that this algorithm does not provide any recovery method when the rate is increased and the next transmission attempt fails. This version of CARA is called CARA-RTS.

Moreover the authors of CARA proposed a mechanism to detect collisions using the Clear Channel Assessment (CCA) when the RTS/CTS handshake is not used: after a Short Inter-Frame Space (SIFS) from the end of data transmission, the algorithm probes the channel through CCA: if the channel is busy but no ACK is received, then a collision is assumed. In this case CARA does not change the counters, as in case of CTS failures.

3.4 Ideal Rate Adaptation Algorithm

In this paper, we use the ideal rate adaptation algorithm as a reference to understand more deeply the performance of AARF-CD. In order to select the rate that maximizes throughput, this algorithm assumes that the sender knows the SNR that characterizes the channel.

It should be noted that, if the RTS/CTS mechanism is used and if the channel is static, this algorithm behaves like RBAR.

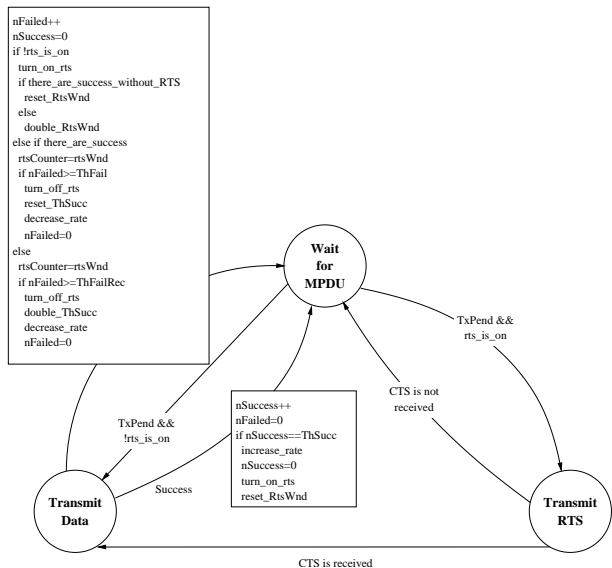


Figure 1. State diagram of AARF-CD

3.5 AARF-CD

AARF-CD is derived from AARF, but it uses the RTS/CTS mechanism only when it is necessary. Figure 1 shows the state diagram of the AARF-CD algorithm and Table 1 defines the parameters used in this paper.

Before the transmission of a packet, AARF-CD checks whether to use the RTS/CTS mechanism or not. This mechanism has to be used when the `rtsCounter` is greater than zero; when it is used and the handshake succeeds (i.e. the CTS is received), this counter is decremented by one. If the CTS is not received, the algorithm retries to send the RTS after the backoff period. If the data frame is successfully transmitted, the `nSuccess` counter is incremented by one; this counter is incremented by one until it reaches the threshold `ThSucc`. At this moment, the next higher rate is used and the RTS/CTS mechanism is turned on setting the `rtsCounter` equal to `RtsWnd`. If the data frame is lost, the algorithm increments by 1 the consecutive failure counter `nFailed`. If the failure occurs without using the RTS/CTS mechanism, RTS/CTS will be turned on by doubling `RtsWnd` if necessary (i.e. when using the current rate transmission attempts fails without the RTS/CTS mechanism). If a new rate is used and the number

Notation	Comments	Value
<code>MinThSucc</code>	Minimum success threshold value	10
<code>MaxThSucc</code>	Maximum success threshold value	60
<code>ThSucc</code>	Number of consecutively success required to increase the rate	[<code>MinThSucc</code> , <code>MaxThSucc</code>]
<code>ThFailRec</code>	Number of consecutive failures required when a new rate is tried in order to decrease the rate	1
<code>ThFail</code>	Number of consecutive failures required in order to decrease the rate	2
<code>nSuccess</code>	Consecutive success counter	[0, <code>ThSucc</code>]
<code>nFailed</code>	Consecutive failure counter	[0, <code>ThFail</code>]
<code>MinRtsWnd</code>	Minimum RTS window value	1
<code>MaxRtsWnd</code>	Maximum RTS window value	40
<code>RtsWnd</code>	Number of consecutively transmissions using the RTS/CTS mechanism	[<code>MinRtsWnd</code> , <code>MaxRtsWnd</code>]
<code>rtsCounter</code>	Consecutive RTS/CTS handshakes counter	[0, <code>RtsWnd</code>]

Table 1. List of parameters used in the AARF-CD code

of unsuccessfully transmission attempts reaches `ThFailRec` (i.e. `ThFailRec` data transmission failures are experienced after the rate increase), the rate is decreased immediately and `ThSucc` is doubled. Otherwise, if RTS/CTS is used and the number of unsuccessfully transmission attempts is greater or equal to `ThFail`, the next lower rate will be used for the next transmission and `ThSucc` is reset to `MinThSucc`. Both `RtsWnd` and `ThSucc` can grow up to a maximum value, called `MaxRtsWnd` for the former and `MaxThSucc` for the latter. The numerical values of these parameters are shown in Table 1.

Note that the RTS/CTS mechanism is switched off every time the rate is decreased and when the `rtsCounter` becomes 0; conversely, it is switched on every time the rate is increased or a data transmission failure occurs without it. Moreover, when a CTS frame is lost, the algorithm does not change the counters `nSuccess`, `nFailed` and `rtsCounter`.

Two main differences between AARF-CD and ARF can be observed: *i*) the adaptation of the number of consecutive transmission attempts needed to increase the rate, and *ii*) the usage of the RTS/CTS mechanism to react to contention problems. While the former is inherited from AARF, the latter is a contribution of this work.

The main difference between AARF-CD and CARA is that, whenever the rate changes, CARA turns off the RTS/CTS mechanism (if the data packet is smaller than a threshold), while AARF-CD switches on the mechanism whenever it increases the rate and it switches off the mechanism when it decreases the rate. The different behavior of AARF-CD guarantees that if the first data transmission attempt fails, it is probably due to channel errors, hence the rate could be decreased immediately. To alleviate this problem, the authors of CARA proposed to detect collision using the CCA. This mechanism works well when there are no hidden terminals, otherwise, it could take wrong decisions. Moreover, when two stations collide, only the station with the shortest transmission time will consider the failure due to collision (e.g. if the colliding stations transmit packets with the same length and with the same

rate, the collision will not be detected by the two stations).

Note that CARA uses fixed thresholds in order to decide both whether to use RTS/CTS and when to decrease or increase the rate. Conversely, AARF-CD uses adaptive thresholds, hence it is more stable when the channel is slowly varying.

4 Performance Evaluation

In order to evaluate the performance of the algorithm, we run simulations using the new NS-3 network simulator. The implementation of ARF, AARF, RRAA and the Ideal rate control algorithms, are distributed within the main release [8], while the code for CARA-RTS and AARF-CD can be found in [11].

We consider an infrastructure scenario with a variable number of nodes, each in the transmission range of the others (i.e. with no hidden terminals) at a variable distance from the AP. All the nodes are equipped with an IEEE 802.11a interface and they use the same rate adaptation algorithm. Each node sends saturated UDP traffic (i.e., the transmission queue is never empty) with a packet size of 2000 bytes without the MAC and PHY headers.

We consider a path loss channel model where the power received in dB is given by

$$P_r = 10 \log(P_t) - n10 \log(d)$$

with P_t , the transmitted power (Watt), d , the distance between the nodes (Meter) and n , an exponential factor equal to 3.

Eight rate adaptation algorithms are compared: Auto Rate Fallback, Adaptive ARF, Robust Rate Adaptation Algorithm, CARA-RTS, AARF-CD, ARF-CD and the Ideal rate adaptation algorithm with and without the RTS/CTS mechanism. ARF-CD is a particular version of AARF-CD where MaxThSucc is equal to 10, i.e. equal to MinThSucc . With these particular settings ARF-CD behaves like a modified version of ARF, since it is not able to adapt the success threshold.

4.1 Single node

As a first step of our analysis, we consider a simple scenario with one node that transmits packets to the AP. During the simulation, the node is moved away from the AP with a step of 1 m. The throughput of the node as function of the distance from the AP (which is proportional to the SNR) is plotted in Figure 2.

As expected, in this scenario ARF and AARF work very well, especially the second algorithm. We can observe that the throughput achieved using AARF is closed to the throughput of the ideal rate adaptation algorithm. The performance of ARF-CD and AARF-CD are very close to the one of ARF and AARF respectively. Hence, in this scenario AARF-CD shows a similar behavior as AARF, as expected. Moreover, the ideal rate

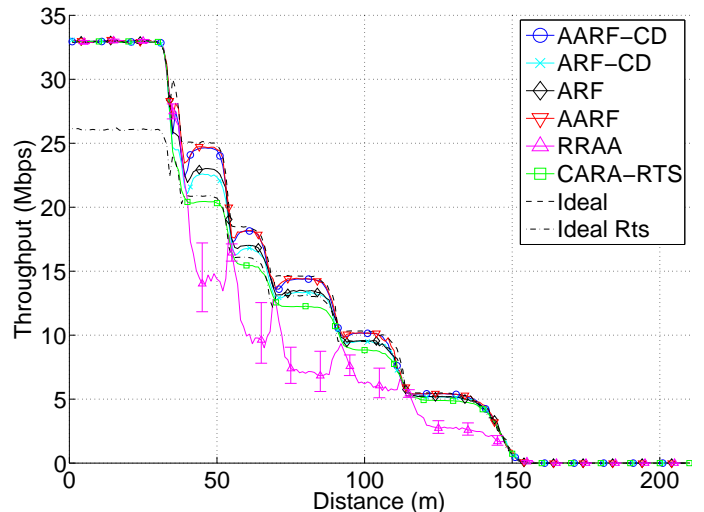


Figure 2. Performance without contention

adaptation algorithm with RTS/CTS (named Ideal RTS) shows that the RTS/CTS mechanism introduces a useless overhead in this scenario. Concordantly AARF-CD does not use RTS/CTS.

RRAA performs always worse than ARF. We can also observe some peaks in the regions corresponding to transitions between two transmission rates. This means that RRAA does not take the best decision for the transmission rate, and it presents instability in the transition regions. To emphasize the instability of RRAA we plot the confidence intervals¹. The parameters used in our simulations are those provided by the authors of [6].

CARA-RTS performs worse than ARF. This is due to the fact that every time a new PHY rate is used, two consecutive transmission failures are required to select again a lower rate.

4.2 Multi-user

Let us now study the behavior of the algorithms in a multi-user scenario. We arranged a variable number of saturated nodes (from 1 to 20), 50 m apart from the AP. Figure 3 shows the aggregated throughput (i.e. the sum of the throughput of all the nodes) as a function of the number of contenders for the eight algorithms.

Performance of AARF-CD and ARF-CD are the closest to the ideal one. As we expected, both ARF and AARF choose a suboptimal rate whenever the number of contending stations is greater than 2. The reason, as we explained before, is that these algorithms assume all packet losses are due to transmission errors. Moreover, we can notice that CARA-RTS is robust against collisions, but it performs worse than AARF-CD and ARF-CD. This is because, when the level of contention increases, CARA-RTS always alternates between basic access

¹We do not plot the confidence intervals of the rate adaptation algorithms different than RRAA because they are small.

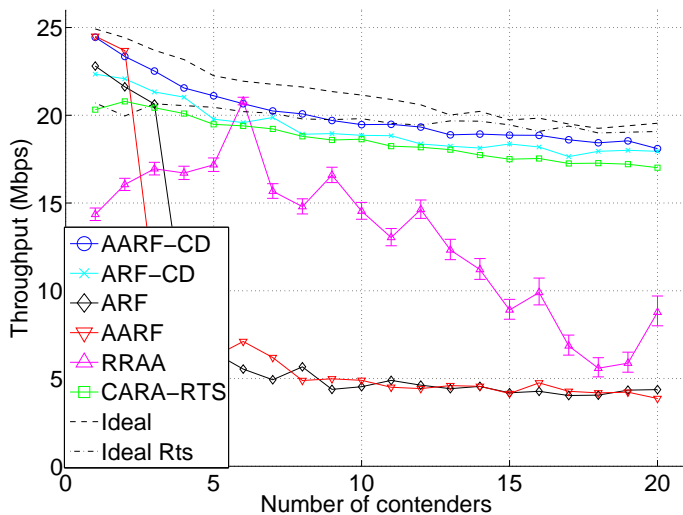


Figure 3. Performance with contention (50m)

and RTS/CTS, while AARF-CD and ARF-CD, increasing the $RtsWnd$ parameter, use predominantly the RTS/CTS mode.

From Figure 3, we observe that RRAA performs poorly with the set of parameters given in [6]. In particular, it is very unstable when the number of nodes increases.

5 Conclusion

In this paper, we presented a novel rate adaptation scheme, called AARF-CD, and we compared it with seven rate adaptation algorithms: ARF, AARF, RRAA, CARA, ARF-CD (which is a particular case of AARF-CD) and an ideal rate adaptation algorithm with an without RTS/CTS handshake. The performance evaluation is done in terms of system throughput, and we show that AARF-CD performs better than ARF, RRAA and CARA, while, in a single user scenario, it performs similarly to AARF.

Simulation results show that, when the number of contenders increase, AARF-CD outperforms the other rate adaptation algorithms and performance obtained are closed to the ideal one.

Finally, we stress that AARF-CD is easy to implement since it is derived from ARF, hence it could be a good alternative to the rate adaptation algorithms already deployed in wireless network devices.

Future work include the simulation comparison of AARF-CD with CARA-BASIC, the use of a more realistic time variant channel model (Jakes), and the performance evaluation when hidden terminals are present in the scenario. Moreover, we will evaluate these algorithms with non-saturated traffic (e.g. VoIP and video streaming). As last step of our analysis, we expect to implement AARF-CD in a testbed and to evaluate its performance in the real world.

References

- [1] IEEE LAN MAN Standards, part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, March 1999.
- [2] IEEE LAN MAN Standards, part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications high-speed physical layer in the 5 ghz band, September 1999.
- [3] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34(4):121–132, 2004.
- [4] A. Kamerman and L. Monteban. WaveLAN-II: A high-performance wireless LAN for the unlicensed band: Wireless. *Bell Labs technical journal*, 2(3):118–133, 1997.
- [5] J. Kim, S. Kim, S. Choi, and D. Qiao. CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs. *IEEE INFOCOM*, 2006.
- [6] Starsky H.Y. Wong, Hao Yang, Songwu Lu, and Vaduvur Bharghavan. Robust rate adaptation in 802.11 wireless networks. In *ACM MOBICOM*, September 2006.
- [7] M. Lacage, M.H. Manshaei, and T. Turletti. IEEE 802.11 rate adaptation: a practical approach. *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 126–134, 2004.
- [8] The Network Simulator - ns-3. Available online at <http://www.nsnam.org>.
- [9] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive MAC protocol for multi-Hop wireless networks. *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 236–251, 2001.
- [10] D. Qiao, S. Choi, and K.G. Shin. Goodput analysis and link adaptation for IEEE 802. 11 a wireless LANs. *IEEE Transactions on Mobile Computing*, 1(4):278–292, 2002.
- [11] AARF-CD and CARA-RTS implementation for ns-3. Available online at <http://telecom.dei.unidp.it/download> “NS2/NS3 section”.