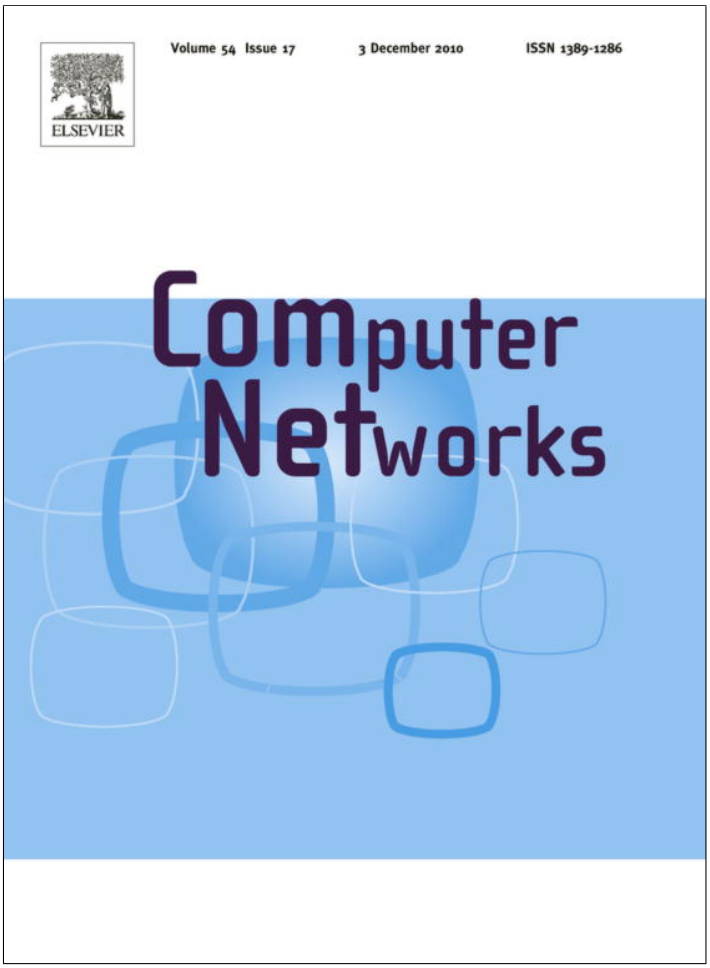


Computer Networks 54 (2010) 3011–3030

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Collision-Aware Rate Adaptation in multi-rate WLANs: Design and implementation

Seongkwan Kim [a,*], Lochan Verma [b], Sunghyun Choi [c], Daji Qiao [d]

[a] System Lab.1, Telecommunication Systems Division, Samsung Electronics Co., Ltd., Suwon 443-742, Republic of Korea
[b] Media Processing Lab., DMC R&D Center, Samsung Electronics Co., Ltd., Suwon 443-742, Republic of Korea
[c] School of Electrical Engineering and INMC, Seoul National University, Seoul 151-744, Republic of Korea
[d] Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, USA

ARTICLE INFO

Article history:
Received 15 December 2009
Received in revised form 31 March 2010
Accepted 23 May 2010
Available online 1 June 2010
Responsible editor: I. Habib

Keywords:
IEEE 802.11 WLAN
Rate adaptation
Collision awareness
ns-2 simulation
MadWifi-based implementation

ABSTRACT

Many rate adaptation algorithms have been proposed for IEEE 802.11 Wireless LAN devices and most of them operate in an open-loop manner, i.e., the transmitter adapts its transmission rate without using the feedback from the receiver. A key problem with such transmitter-based rate adaptation schemes is that they do not consider the collision effect. Accordingly, they often result in severe throughput degradation when many transmission failures are due to frame collisions. In this paper, we present a transmitter-based rate adaptation scheme, called CARA (Collision-Aware Rate Adaptation), and its MadWifi-based implementation. The key idea of CARA is that the transmitter combines adaptively the RTS/CTS (Request-to-Send/Clear-to-Send) exchange with the CCA (Clear Channel Assessment) functionality in order to differentiate frame collisions from transmission failures due to channel errors. The effectiveness of CARA schemes is evaluated via extensive ns-2 simulations and testbed experimentations.

1. Introduction

The transmission rate adaptation has gained interests as for a dominant issue to enhance the performance of IEEE 802.11 WLAN (Wireless LAN) technology. While the 802.11 PHYs (Physical layers), e.g., 802.11a/b/g, provide a multi-rate feature, the standard does not specify any algorithm and/or protocol to utilize them efficiently. In the past few year, many rate adaptation schemes have been proposed in the literature [1–17].

The effectiveness of a rate adaptation scheme depends on how fast it can respond to the variation of the wireless channel condition. Moreover, since multiple stations often compete for the shared wireless medium in an 802.11 system, frame collisions are inevitable due to the contention nature of the 802.11 DCF (Distributed Coordination Function). Therefore, the effectiveness of a rate adaptation scheme also depends greatly on how the collisions are detected and handled.

In a rate adaptation scheme, a transmitter may adapt its transmission rate with or without using the feedback from the receiver, where the feedback information could be either SINR (Signal to Interference and Noise Ratio) or the transmission rate desired by the receiver. Depending on whether to use the feedback from the receiver, rate adaptation schemes can be classified into two categories: *receiver-based*, i.e., with the feedback, and *transmitter-based*, i.e., without the feedback. Unfortunately, most existing transmitter-based schemes malfunction severely when there are many contending stations in the network, because they can not differentiate frame collisions from transmission failures caused by channel errors. This results in decreasing the transmission rate over-aggressively due to many collision-induced transmission failures. For example, the widely-adopted ARF (Automatic Rate Fallback) scheme

* Corresponding author. Tel.: +82 1088704000.
E-mail addresses: seongkwan@ieee.org (S. Kim), lochan.verma@samsung.com (L. Verma), schoi@snu.ac.kr (S. Choi), daji@iastate.edu (D. Qiao).

1389-1286/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2010.05.013

[1,2] does not work properly in multi-user environments because it decreases the transmission rate upon consecutive frame collisions, which has been reported in [18] based on both simulation and empirical results. In contrast, the collision effect is mitigated with receiver-based schemes, such as RBAR (Receiver-Based Auto Rate) [14] and OAR (Opportunistic Auto Rate) [15], thanks to the interaction between the transmitter and the receiver.

In this paper, we design a transmitter-based rate adaptation scheme with collision awareness, called CARA (Collision-Aware Rate Adaptation). The key idea of CARA is that the transmitter combines adaptively the RTS/CTS (Request-to-Send/Clear-to-Send) exchange with the CCA (Clear Channel Assessment) functionality to differentiate frame collisions from channel-error-caused transmission failures. Considering its wide adoption in many commercial 802.11 devices, ARF is chosen as the baseline rate adaptation scheme for CARA. CARA specifies three different methods: (i) CARA–RTS that identifies collision via RTS Probing and makes collision-aware rate-decrease decisions; (ii) CARA–CCA that identifies collision via RTS Probing as well as CCA Detection, based on which the rate-decrease decisions become enhanced; and (iii) CARA–RI that makes collision-aware rate-increase decisions while identifying collision via CARA–CCA. The preliminary results of CARA were originally presented in [16]. At that time, CARA was the first effort in applying adaptive usage of RTS/CTS exchange and CCA functionality to differentiate frame collisions from channel-error-caused transmission failures. Since then, a few rate adaptation schemes have been proposed to improve upon CARA, such as RRAA (Robust Rate Adaptation Algorithm) in [11] and PBRA (Probabilistic-Based Rate Adaptation) in [12]. Nonetheless, they are all designed based on the idea of adaptive RTS/CTS that is similar to that in CARA. In this paper, we present the complete details of the CARA design and describe its implementation on the MadWifi platform [4].

In particular, we make the following contributions.

- CARA has the originality in rate selection with collision awareness in IEEE 802.11a/b-based wireless networks.
- CARA works in a purely transmitter-based manner. No feedback information from the receiver is required in its collision-aware rate decisions.
- We provide the complete view of CARA from its algorithmic design to the implementation details, running upon a practical device platform.
- The effectiveness of CARA has been thoroughly evaluated, considering various aspects: (1) constant vs. varying channel qualities; (2) static vs. mobile stations; (3) single vs. multiple contending stations; (4) non-hidden vs. mutually hidden stations; (5) one-hop vs. multi-hop topological settings; (6) simulation vs. real testbed environments; and (7) superiority check to many existing algorithms.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. Section 3 briefly overviews the IEEE 802.11 MAC, the RTS/CTS exchange in IEEE 802.11, and the ARF scheme. The details of the CARA design are described in Section 4, followed by Section 5 that presents in-depth simulation study on CARA. Section 6 discusses the CARA implementation on the MadWifi platform and experimental results. Finally, the paper concludes in Section 7.

2. Related work

In receiver-based schemes [14,15], after the receiver specifies its desired transmission rate and feeds it back to the transmitter as part of the modified RTS/CTS exchange, the transmitter adjusts its transmission rate accordingly. Since the rate adaptation is dictated by the receiver, rate selection decisions are not affected by frame collisions. However, in order to support such a feedback loop, the CTS (and possibly RTS) frame format should be modified to convey the extra information that does not conform to the 802.11 standard. Moreover, there should be a predetermined reference table, e.g., SINR vs. FER (Frame Error Rate), based on which the receiver calculates and feeds back the most appropriate transmission rate to the transmitter. The reference itself might not become universal as the error performance varies over devices [13]. In practice, using the RTS/CTS exchange could be a costly solution that could waste the precious wireless bandwidth when hidden stations do not exist. It should be noted that the RTS/CTS exchange is rarely used in practical infrastructure-based WLANs due to this fact.

With transmitter-based schemes, a transmitter makes the rate adaptation decision solely based on its local information. Since they do not require any interaction between the transmitter and the receiver, transmitter-based schemes are standard-compliant in general. Transmitter-based schemes can be further classified into two subcategories. The first subcategory decides the transmission rate based on local channel estimation [7–13].

The schemes in this subcategory often yield good performance similar to that of closed-loop approaches, but usually require extra implementation efforts; either being equipped with preexamined Refs. [7–9,13] or being able to estimate loss/throughput over the target link [10–12]. In contrast, the second subcategory only makes use of the local Ack (Acknowledgment) information when selecting the transmission rate [1–6], which is very simple to implement. This is also the main reason why the ARF algorithm and its variants, belonging to the second subcategory, is adopted by many commercial 802.11 WLAN products.

It has been pointed out in [6] that there are two fundamental issues when designing a rate adaptation scheme, i.e., *when to increase* and *when to decrease* the transmission rate. The effectiveness of a rate adaptation scheme depends greatly on how fast it can respond to the variation of the wireless channel condition. The schemes presented in [3,6] address the first issue and enhance the original ARF by allowing a transmitter to increase its rate in an adaptive manner over a time-varying wireless channel. In [16], we studied the effect of frame collisions on the second issue, i.e., *when to decrease* the transmission rate and introduced collision awareness into rate-decrease decisions. We also found that frame collisions may as well pose negative impact on *when to increase* the transmission rate; the more

contending stations in the network hence the higher frame collision probability, the more delayed rate increase will be. To deal with such a delayed rate-increase problem, we propose CARA–RI in this paper by introducing collision awareness into rate-increase decisions.

There are a few other papers that deal with the incorrect rate adaptation problem due to frame collisions. RRAA proposed in [11] uses RTS frames more aggressively than CARA with an adaptive *RTSwnd* parameter. RTSwnd represents the number of data frames to be transmitted with RTS support and it varies as follows. It increases linearly when a data transmission without RTS support fails, and decreases in a multiplicative manner when the cause of a data transmission success or failure is clear, e.g., when a data transmission with RTS support succeeds (implying no channel errors and no frame collision) or when a data transmission with RTS support fails (which must be due to channel errors). PBRA in [12] also utilizes RTS/CTS frames to resolve the collision problem, yet in a different way compared with CARA and RRAA. The activation of the RTS/CTS exchange is decided based on a probability P^*_{rts} that minimizes the expected time to successfully transmit the current data frame. To have a correct value of P^*_{rts}, a station with PBRA estimates the collision probability based on the result of each transmission attempt and then, it uses a mathematical model to calculate P^*_{rts}. Although RRAA and PBRA might make a better decision than CARA in terms of rate adaptation, it should be noted that those schemes were proposed after CARA had been introduced in the literature. Moreover, the rule to activate the RTS/CTS exchange of RRAA has a clear similarity to that of CARA, i.e., sending an RTS frame when the previous data transmission failed.

3. System overview

3.1. IEEE 802.11 CSMA/CA

The mandatory MAC scheme, DCF specified in IEEE 802.11 standard [19] is based on CSMA/CA as illustrated in Fig. 1a. Even with random backoff, a transmitted frame may still collide with other frames when two or more stations finish the backoff at the same time. Such frame collisions cannot be completely eliminated due to the contention nature of DCF, and the problem becomes worse as the number of contending stations increases. Besides collision, a frame transmission failure may also be caused by the deteriorated channel condition.

When there are hidden stations in the network, the performance of the basic CSMA/CA may degrade severely. The unprotected time interval due to frame collisions however, can be shortened to the RTS transmission time by preceding the data frame transmission with the exchange of two short control frames, i.e., RTS and CTS frames, and hence, the hidden-station problem may be ameliorated. This is known as the original design purpose of the RTS/CTS exchange that is illustrated in Fig. 1b. In most 802.11 devices operating in infrastructure-based WLANs with APs (Access Points), however, the RTS threshold is set to the largest value, i.e., 2347 octets, which basically disables the usage of RTS/CTS exchange.

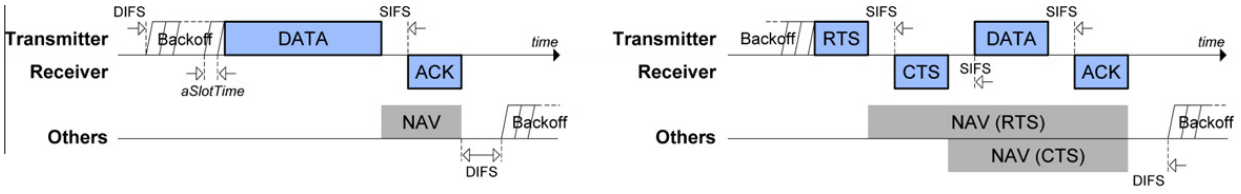
It is also known that the RTS/CTS exchange is useful in heavily contending WLAN environments, where many transmissions may fail due to collisions, and the benefit is particularly salient with large data frame sizes [20]. The author of [20] also pointed out that the throughput performance with RTS/CTS is less sensitive to the number of contending stations and the collision probability than that without RTS/CTS.

In this paper, we propose an adaptive usage of the RTS/CTS exchange as a means to probe the channel state in order to differentiate frame collisions from channel-error-caused transmission failures. Since the support of the RTS/CTS exchange is a mandatory part of the 802.11 standard, our approach can be implemented with commercial 802.11 devices, which will be demonstrated in Section 6.

3.2. ARF in IEEE 802.11

In an early packet-radio network design, an Ack-based rate selection scheme was proposed in [21]. Considering that poor channel quality may cause unsuccessful transmission of data and Ack frames, the authors of [21] proposed the idea of lowering the transmission rate when a certain number of expected Ack frames have been lost. Similarly, the presence of consecutive Ack frame receptions can be deemed an indication of improved channel quality, which may be good enough to support a higher transmission rate.

In the 802.11 system, a similar Ack-based rate adaptation scheme was proposed in [1,2], i.e., ARF, which was originally developed for Lucent Technologies' WaveLAN-II devices. It has been widely adopted and implemented in many commercial 802.11 devices since then. ARF alternates the transmission rates by keeping track of a timing



(a) Basic channel access mechanism.

(b) RTS/CTS channel access mechanism.

Fig. 1. Two channel access mechanisms of the IEEE 802.11 DCF. SIFS (Short Inter-Frame Space) is the smallest time interval used between two consecutive frame transmissions.

function as well as missing Ack frames. If two consecutive Acks are not received correctly by the sender, the second retry of the data frame and subsequent transmission attempts are done at a lower transmission rate and a timer is started. When either the timer expires or the number of successfully received Acks reaches 10, the transmission rate is raised to the next higher level and the timer is canceled. However, if an Ack is not received for the very next transmission attempt, the transmission rate is lowered again and the timer is restarted. Such an expediting rate decrease with unjustified rate increase decisions is referred to as *probation state* [2].

Apparently, due to its heuristic and conservative nature, ARF does not react quickly when the wireless channel condition fluctuates. In other words, the transmitter may attempt increasing its transmission rate to probe the wireless channel condition upon consecutive successful Ack receptions and decreasing its rate upon consecutive (re) transmission failures without considering the actual cause of transmission failures (i.e., channel errors or frame collisions). However, because of its simplicity, ARF is still widely employed in commercial 802.11 devices, and many proposed transmitter-based rate adaptation schemes [3–6] are rooted in ARF.

4. CARA: Collision-Aware Rate Adaptation

In this section, we present the design details of CARA. CARA specifies three different methods to differentiate frame collisions from channel-error-caused transmission failures: (i) CARA–RTS that identifies collisions via RTS Probing and makes collision-aware rate-decrease decisions; (ii) CARA–CCA that identifies collisions via RTS Probing as well as CCA Detection, based on which the rate-decrease decisions become enhanced; and (iii) CARA–RI that makes collision-aware rate-decrease/-increase decisions while identifying collisions via CARA–CCA.

4.1. CARA–RTS: introducing collision-aware rate decrease via RTS Probing

RTS Probing is the mandatory collision differentiation method in CARA. The transmission error probability of RTS and CTS frames is relatively small because of its small sizes (20/14 bytes) and robust transmission rate (one of the basic rate set). We assume that an RTS frame activated for the purpose of CARA's channel probe is transmitted at the lowest rate, and hence, transmission failures of RTS happen mostly due to collisions. On the other hand, we know that a data transmission failure following a successful RTS/CTS exchange is due to channel errors in most cases, because the successful RTS/CTS exchange has prevented other stations accessing the wireless channel, and thus reduces the collision probability of the subsequent data transmission significantly. Therefore, if we exchange RTS/CTS frames before each data transmission and then apply the ARF scheme, misinterpretation of a data frame collision as a channel-error-caused data transmission failure is less likely to happen. As a result, unnecessary rate decrements could be avoided. One side effect of this approach is the added RTS/CTS overhead that wastes the precious wireless bandwidth. In fact, the RTS/CTS option is disabled in most 802.11 products currently available in the market. Based on the above observations, instead of mandating an RTS/CTS exchange before each data frame transmission, we propose RTS Probing that enables RTS/CTS exchange only when a data frame transmission fails.

4.1.1. State transition diagram

The detailed procedure of RTS Probing is best explained with a transmitter's state transition diagram as shown in Fig. 2a, where related symbols and parameters are listed in Table 1. There are four states in the diagram:

- *Initial State*: the starting point of the procedure;
- *Wait for MPDU*[1]: a station is in this state when there are new data frames coming from the upper layer or when the current frame transmission fails and retransmission has been requested;
- *DATA Tx*: a station is in this state when it finishes a data transmission and waits for the corresponding Ack;
- *RTS Tx*: a station is in this state when it finishes an RTS transmission and waits for the corresponding CTS.

As shown in Fig. 2a and Table 1, the consecutive failure count (n) is compared with two different thresholds, the probe activation threshold (P_{th}) and the consecutive failure threshold (N_{th}), for different purposes. When n reaches P_{th}, the RTS/CTS frames will be exchanged before the next data retransmission attempt, while when n reaches N_{th}, the next data retransmission attempt will be conducted at a lower rate. With different values for P_{th} and N_{th}, the RTS Probing procedure works differently, and the default values in CARA are 1 and 2 for P_{th} and N_{th}, respectively. Some example scenarios are listed below:

- $P_{th} = 0$: in this case, the RTS/CTS frames are exchanged before each data (re) transmission attempt. When an RTS/CTS exchange succeeds, the data frame is (re) transmitted. The data transmission rate falls back to the next lower level, if available, upon N_{th} data transmission failures.
- $P_{th} \geqslant 1$, $N_{th} = 1$: in this case, a data frame is transmitted without RTS/CTS support, and its rate falls back upon a single transmission failure. Note that the RTS/CTS exchange is never activated when $P_{th} \geqslant N_{th}$.
- $P_{th} \geqslant 2$, $N_{th} = 2$: in this case, the rate falls back upon two consecutive transmission failures without RTS/CTS support. This is equivalent to ARF if the consecutive success threshold (M_{th}) is set to 10.
- $P_{th} = 1$, $N_{th} = 2$ (the default values in CARA): in this case, a data frame is first transmitted without RTS/CTS support. If the transmission fails, the RTS/CTS exchange will be activated for the next retransmission attempt, and the transmission rate falls back if retransmission fails again.

[1] In the nomenclature of IEEE 802.11 [19], MPDU (MAC Protocol Data Unit) that is interchangeably used with *frame* is the data unit exchanged between two peer MAC entities. MSDU (MAC Service Data Unit) refers to the MAC payload.

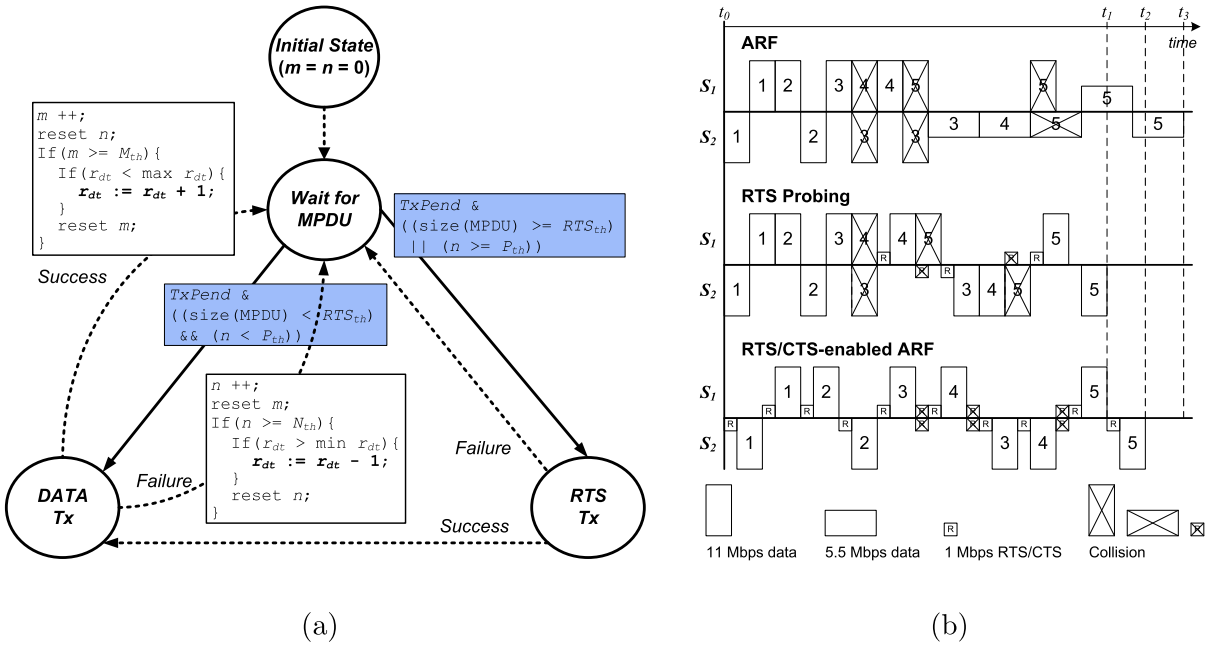


Fig. 2. (a) The state transition diagram of RTS Probing and (b) an illustrative example of ARF, RTS/CTS-enabled ARF, and RTS Probing timelines for a two-station 802.11b network, where the channel state is good enough to accommodate the highest rate of 11 Mbps.

Table 1
List of notations used to describe CARA.

Notations	Comments
m	consecutive success count
n	consecutive failure count
M_{th}	consecutive success threshold
N_{th}	consecutive failure threshold
P_{th}	probe activation threshold
TxPend	status: a data frame is pending
R_{dt}	array of transmission rates (in Mbps) 802.11a = {6, 9, 12, 18, 24, 36, 48, 54}, 802.11b = {1, 2, 5.5, 11}
r_{dt}	rate index: an element of R_{dt}
$\min r_{dt}$/ $\max r_{dt}$	min/max indexes of r_{dt}
RTS_{th}	frame size-based RTS Threshold as defined in the standard

It is interesting to see that, if the wireless channel condition suddenly becomes so bad that an RTS transmission fails, the transmitter may be stuck in the ⟨Wait for MPDU ↦ RTS Tx ↦ Wait for MPDU⟩ loop. Fortunately, since data frames are typically more vulnerable to channel errors than RTS frames, it is not likely that data frames could be transmitted successfully in this situation. Therefore, there is little undesired side effect caused by the existence of such a loop. Once the wireless channel recovers from the bad state and after an RTS frame is delivered successfully, the data frame transmission attempts may resume. M_{th} in the state transition diagram represents the number of consecutive successful frame transmissions that a transmitter needs to observe before increasing its transmission rate. In CARA, we set M_{th} to be the same as in ARF: $M_{th} = 10$. In fact, CARA schemes may be combined with the schemes [3,6] that adjust the M_{th} value to the time-varying wireless channel condition to achieve even better performance.

4.1.2. Example

We use a simple example to illustrate the RTS Probing procedure and compare it with ARF and RTS/CTS-enabled ARF in Fig. 2b. Assume that two 802.11b stations S_1 and S_2 are contending for the shared wireless medium with the same data frame size (1500-byte MSDU), and the channel condition is good enough to accommodate the highest transmission rate of 11 Mbps. Then, the data/Ack exchange duration at the 11 Mbps data transmission rate ($\text{data}_{11\,\text{Mbps}} \mapsto$ SIFS (Short Inter-Frame Space) $\mapsto \text{Ack}_{2\,\text{Mbps}}$) is approximately twice the RTS/CTS exchange duration ($\text{RTS}_{1\,\text{Mbps}} \mapsto \text{SIFS} \mapsto \text{CTS}_{1\,\text{Mbps}} \mapsto \text{SIFS}$). Similarly, the data/Ack exchange duration at the 5.5 Mbps data transmission rate ($\text{data}_{5.5\,\text{Mbps}} \mapsto \text{SIFS} \mapsto \text{Ack}_{2\,\text{Mbps}}$) is about four times the RTS/CTS exchange duration. Fig. 2b reflects the relative differences of these durations. Successful transmissions are shown as empty rectangles in the figure, while crossed rectangles represent frame collisions. We assume that stations are transmitting frames in a saturated mode, i.e., their data queues are never empty. Therefore, the collision probability of a frame transmission remains the same regardless of the type of the frame. We omit inter-frame spaces, backoff durations, and Ack transmissions for clarity.

It is clear that with the ARF scheme, after four data transmission attempts, S_2's transmission rate falls back to 5.5 Mbps, while with RTS Probing, S_2 is still able to preserve the highest transmission rate at 11 Mbps. Moreover, we can see that with RTS Probing, five frame transmissions are completed at an earlier time than that when ARF is used (i.e., $t_1 < t_3$), meaning that more frame transmissions may be accommodated when using CARA with RTS Probing

and hence, better channel utilization may be achieved. For the comparison purpose, we also show the timeline of transmission rate selection for the RTS/CTS-enabled ARF scheme in the figure. Though this scheme does not suffer from incorrect rate decreases as the ARF scheme, it finishes frame transmissions later than RTS Probing (i.e., $t_1 < t_2$) due to the increased overhead of always-enabled RTS/CTS exchange. In this paper, we refer to the CARA scheme with RTS Probing only as CARA–RTS.

4.2. CARA–CCA: enhanced collision-aware rate decrease via RTS Probing + CCA Detection

The second method to differentiate frame collisions from channel errors is *CCA Detection. CCA as specified in the standard is the mechanism for the PHY to determine whether the channel is busy or idle.* Before describing the details of CCA Detection, let's take a look at Fig. 3 that illustrates two possible collision scenarios under the assumption of no hidden stations. In this figure, S_1 is the transmitter of our interest while S_2 represents another station that starts its frame transmission simultaneously with S_1. Actually, this figure could also represent multiple frame collisions, and in such a case, S_2 represents the station transmitting the frame with the longest transmission time among colliding frames. Collisions can be classified into two cases as shown in Fig. 3: in Case 1 (Case 2), the transmission duration of S_1's data frame is longer (shorter) than that of S_2's data frame.

In general, the CCA function in the 802.11 is used by a wireless station to assess the channel occupancy state at a given time. In the case of CCA busy, the station freezes its backoff process. In the case of CCA idle, the backoff process may be resumed or the station may start transmitting if the backoff count reaches zero. Our CCA Detection method works as follows. At SIFS time after a transmitter finishes its data transmission, it starts assessing the wireless channel using CCA. Since the station expects an Ack reception at this time point, so if the wireless channel is assessed as busy while the expected Ack reception does not start, the station concludes that a collision has just occurred to its data transmission, which corresponds to Case 2 in Fig. 3. In this situation, since a frame collision has already been identified successfully by CCA Detection, RTS Probing will not be launched; hence, the overhead induced by the RTS/CTS exchange can be avoided. The transmitter would then retransmit without increasing the failure count (n) or lowering the transmission rate.

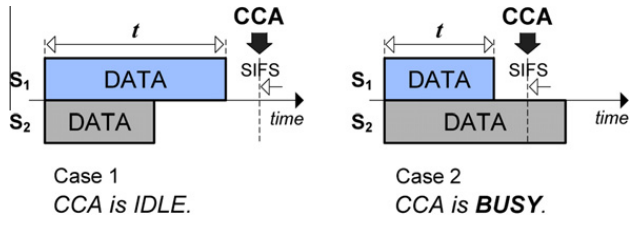


Fig. 3. Two possible cases of collision with no hidden stations, where Case 2 can be detected by CCA Detection.

On the other hand, CCA Detection is not able to identify a frame collision in Case 1. Note that Fig. 3 illustrates possible collision cases without hidden stations; CCA Detection may not work for a collided frames that are transmitted from stations being mutually hidden from one another with which a collision may happen in the middle of a data transmission. In such situations, RTS Probing will be launched after the transmission failure. We refer to the CARA scheme that uses both RTS Probing and CCA Detection as described above as CARA–CCA.

4.3. CARA–RI: extending collision awareness to rate-increase decisions

CARA–CCA yields higher aggregate system throughput than ARF by making collision-aware rate decrease decisions using RTS Probing and CCA Detection. However, when attempting to increase the transmission rate, it simply employs ARF's counting algorithm and resets the consecutive success count (m) upon a frame transmission failure. To further improve the system performance, we introduce collision-aware rate increase into CARA–CCA, and the resulting scheme is referred to as CARA–RI. Before presenting the details of CARA–RI, we first demonstrate the problems with ARF's rate-increase algorithm using simulation results.

We use the ns-2 [22] to simulate a star-topology 802.11b network with five stations evenly spaced on a circle around the AP with the radius of 40 m. Fig. 4 illustrates how one of the stations adapts its transmission rate (with ARF) to the fluctuation of the wireless channel condition for 1 s. In this figure, the left y-axis represents the SNR (Signal to Noise Ratio), whose variation is plotted as the solid curve, and the right y-axis shows the available 802.11b transmission rates. The left y-axis shows three real numbers instead of being evenly graduated. They stand for the threshold values of SNR over which a station should change its transmission rate to maximize the single-station throughput. Three types of events may occur after each data transmission attempt and different types of events are illustrated with different symbols in the figure:

- +: a successful data frame transmission;
- □: a data transmission failure caused by channel error;
- △: a data transmission failure due to collision.

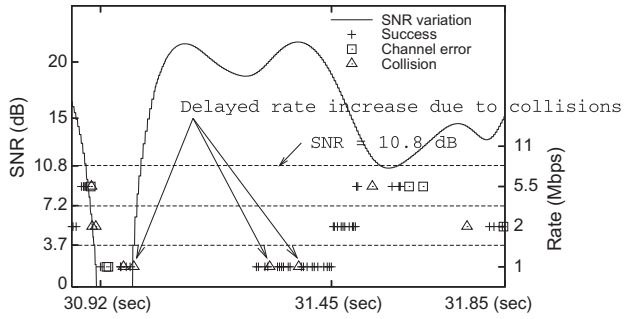


Fig. 4. ARF's rate adaptation when five 802.11b stations are contending in a time-varying fading channel.

We have two observations from Fig. 4. First, ARF cannot differentiate frame collisions from channel errors, thus decreasing the transmission rate over-aggressively. For example, we observe that ARF decreases the rate from 5.5 to 2 Mbps after two consecutive transmission failures that are induced by frame collisions. Therefore, it operates at the low rates of 1 or 2 Mbps during most of the observed period. This problem can be dealt with in CARA–CCA. Second, we observe *delayed rate increase attempts* due to frame collisions. For example, between 30.92 and 31.45 s, we observe four frame collisions (shown as " △" points) that cause the station to delay its rate-increase attempts and to remain operating at the lowest rate of 1 Mbps. This is clearly not desirable because, according to the employed channel-error model, the SNR value of 10.8 dB (shown as the first dashed line from top of the figure) is high enough to accommodate the highest transmission rate of 11 Mbps.

Based on the above observations, we propose CARA–RI as an extension to CARA–CCA, which makes collision-aware rate-increase decisions. Specifically, unlike in ARF and CARA–CCA, where the consecutive success count (m) is reset with any frame loss, CARA–RI only resets m when the frame loss occurs with RTS/CTS preceding the transmission attempt. In other words, m is reset only when the frame loss can be identified as a channel-error-caused failure that is done by comparing n with P_{th}. As a result, CARA–RI adjusts the transmission rate more quickly to the improving wireless channel condition than CARA–CCA.

Fig. 5 shows the pseudo-code of CARA–RI and related notations are listed in Table 1. With the default setting, i.e., $P_{th} = 1$, $N_{th} = 2$, and $M_{th} = 10$, CARA–RI works as follows. A data frame is first transmitted without RTS/CTS support (Line 7). After the duration of data transmission (t) plus SIFS time interval, the station employs CCA to assess the wireless channel (Line 7). In the case of CCA busy, the station retransmits the frame without increasing n (Line 15). If the transmission fails and the wireless channel is not assessed as busy, n increases and then the RTS/CTS exchange will be activated for the next retransmission attempt (Lines 2–3), and the transmission rate falls back if the retransmission fails again (Lines 13–19). Note that CCA Detection is only applied when $n < P_{th}$ and $\ell < RTS_{th}$, i.e., before RTS Probing is launched. Moreover, the consecutive success count (m) is compared with the consecutive success threshold (M_{th}) and increases upon a successful frame transmission (Lines 8–12). m is reset only when the frame loss can be identified as a channel-error-caused failure (Lines 14 and 19). In comparison, with CARA–RTS and CARA–CCA, m is reset whenever a data transmission fails.

5. Performance evaluation via simulation study

We conduct diverse simulation runs using the ns-2. In order to consider many different and realizable scenarios, which may not be straightforwardly evaluated in a real testbed, our simulation study includes multiform topological settings with the following factors: distribution of stations, number of nodes, existence of hidden stations, mobility, channel quality, and number of hops. After describing the details for the simulation setup, we show evaluation results and the corresponding analysis. The evaluation starts assuming an AWGN (Additive White Gaussian Noise) channel and then we realize the wireless medium with a multi-path fading channel.

5.1. Simulation setup

The 802.11 DCF module in the ns-2 simulator is enhanced to support CARA–RTS/-CCA/-RI and ARF. The 802.11b PHY and the time-varying fading channel model are also realized in the simulator. We use the empirical BER (Bit Error Rate) vs. SNR curves that are provided by Intersil,[2] to estimate the FER [23]. Each station transmits with 20 dBm power and the background noise level is set to −96 dBm. Besides, we use a log-distance path-loss model with the path-loss exponent of four to simulate the indoor office environment [24]. We use the Ricean fading model to simulate the time-varying wireless channel condition [25]. The Ricean distribution is given by:

$$p(r) = \frac{r}{\alpha^2} e^{\left(-\frac{r}{2\sigma^2}+K\right)} I_0(2Kr),$$

where K is the distribution parameter representing the line-of-sight component of the received signal,[3] r is the received power, α^2 is the variance of the background noise, and $I_0(\cdot)$ is the modified Bessel function of the first kind and zero order [24]. The frequency of the channel condition variation depends on the relative speed of the mobile station to its surroundings. Note that the channel fluctuation may also occur to static stations because of the moving environment. We assume 2.5 m/s speed of the moving environment in our simulation that corresponds to the Doppler spread of 20 Hz.

We evaluate the following testing schemes: (i) constant-rate schemes (referred to as Rx) using fixed rate of x Mbps; (ii) ARF; (iii) ARF using the RTS/CTS exchange all the time (referred to as ARF–RTS/CTS); (iv) CARA–RTS (with RTS Probing only); (v) CARA–CCA (with both RTS Probing and CCA Detection); (vi) CARA–RI (with CARA–CCA and collision-aware rate increase); and (vii) Genie (an ideal rate adaptation scheme that makes perfect rate selection decisions and a proper usage of the RTS/CTS exchange, thus yielding the highest achievable throughput in a given environment). The testing schemes are compared with each other in terms of the aggregate system throughput (in Mbps or the number of frames that were successfully transmitted). As addressed in Section 4, we set the consecutive success threshold (M_{th}) to 10 and the consecutive failure threshold (N_{th}) to 2 in both ARF and CARA schemes. Moreover, we set the probe activation threshold (P_{th}) to 1 in CARA schemes. RTS/CTS frames are always transmitted at the lowest rate of 1 Mbps. Each station transmits in a saturated mode and all the data frames are transmitted without fragmentation. We use LLC/IP/UDP as the upper layer protocol suite, and the MSDU length is 1500 octets unless specified otherwise. *The num-*

[2] The BER curves in [23] were measured in an AWGN environment without fading.

[3] For $K = 0$, the Ricean distribution reduces to the Rayleigh distribution, in which there is no line-of-sight component.

INITIALIZATION:

$\mathcal{F}$:= the frame at the head of the data queue; ℓ := the frame size of $\mathcal{F}$; $m := 0$; $n := 0$; t := the frame transmission time of $\mathcal{F}$; $r_{dt} := \min r_{dt} = 0$; $\max r_{dt}$:= the number of available rates $- 1$;

LOOP:

```
1.  while (the data queue is not empty)
2.      if (ℓ ⩾ RTS_th ‖ n ⩾ P_th)
3.          ⟨ an RTS frame is transmitted ⟩;
4.          if (a CTS frame is correctly received)  ⟨ F is transmitted ⟩;
5.          else  continue;
6.      else
7.          ⟨ F is transmitted ⟩;  ⟨ do CCA after (t + SIFS) ⟩;
8.      if (an Ack frame is correctly received)
9.          reset n;  m := m + 1;
10.         if (m ⩾ M_th)
11.             if (r_dt < max r_dt)  r_dt := r_dt + 1;
12.             reset m;
13.     else
14.         if (n ⩾ P_th)  reset m;
15.         if (CCABusy == TRUE)  continue;
16.         n := n + 1;
17.         if (n ⩾ N_th)
18.             if (r_dt > min r_dt)  r_dt := r_dt − 1;
19.             reset m, n;
```

Fig. 5. Pseudo-code of CARA–RI.

ber of stations varies over simulation scenarios and is stated in each case.

For an end-to-end routing in the case of multi-hop communication scenarios, we employ ETT (Expected Transmission Time) metric to estimate link quality [26]. ETT is the time spent in transmitting a data frame and originates from ETX (Expected Transmission Count) [27], which is an early generation of link metric for wireless mesh routing. ETT is calculated using the following formula; $\text{ETT} = \text{ETX} \times \frac{\ell}{r}$, where ℓ denotes the nominal size of data frames and r is the transmission rate used over the target link. ETX presented above also follows the original design, i.e., $\text{ETX} = \frac{1}{d_f \times d_b}$, where d_f and d_b are measured with hello messages transmitting at the lowest transmission rate. An end-to-end route setup is done in an optimal manner. For given ETT values of all available wireless links, the routing protocol searches the optimal path such that the CETT (Cumulative ETT) of the selected path becomes the minimum, which means that an end-to-end frame transfer delay is expected to be the minimum. We omit detailed descriptions about routing that is out of scope of this paper.

5.2. AWGN channel environments

In the first part of the simulation with the AWGN channel (Sections 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6), all stations are set to be static to clearly demonstrate the overhead and the rate adaptability of testing schemes.

5.2.1. One-to-one topology

We compare the testing schemes in the simplest two-station one-hop topology, in which one station transmits continuously to the AP with various distance of r ($30 \leqslant r \leqslant 80$) m. Simulation results are plotted in Fig. 6a.

In general, the throughput decreases for all testing schemes as the distance increases. R1 is the most conservative scheme of all. It transmits all the frames at the lowest 1 Mbps, thus resulting in the lowest throughput when r is small. At the same time, due to its strong error-correcting capability, even when the transmitter is far away from the receiver, they are still able to communicate successfully. On the other hand, R11 is the most aggressive scheme that transmits all the frames at the highest 11 Mbps. R11 allows the transmitter to make better use

of the available bandwidth when r is small. However, due to its poor error-correcting capability, the throughput degrades drastically as r increases. In fact, when $r > 47$, all the transmission attempts fail and the throughput drops to zero. Other constant-rate schemes can be viewed as compromises between R1 and R11.

Note that there is no contention in this topology; hence, one of the constant-rate schemes is supposed to perform the best for a given distance. Genie always yields the highest achievable throughput. That is, its throughput curve matches the outer envelop of those of the constant-rate schemes. We observe that CARA–RTS performs close to Genie for the entire range of the simulated distance. Moreover, thanks to its adaptive activation of the RTS/CTS exchange, CARA–RTS achieves comparable throughput with ARF. In comparison, ARF–RTS/CTS yields much lower throughput than both ARF and CARA–RTS due to added overhead of RTS/CTS exchange before each data transmission attempt, which is consistent with our earlier observations in Fig. 2b.

5.2.2. Star topology

We next consider star-topology networks with varying number of contending stations in order to study the collision effect on the system performance. In this simulation, various number of contending stations are evenly spaced on a circle around the AP with the radius of 40 m. Note that, according to the previous results in one-hop topology networks, a station at $r = 40$ m away from the AP shall use 11 Mbps, if it is equipped with an ideal rate adaptation scheme (e.g., Genie). As shown in Fig. 6b, CARA–RI shows 51.1%, 15.0% and 5.7% (on average) throughput improvements over ARF, ARF–RTS/CTS, and CARA–RTS, respectively.

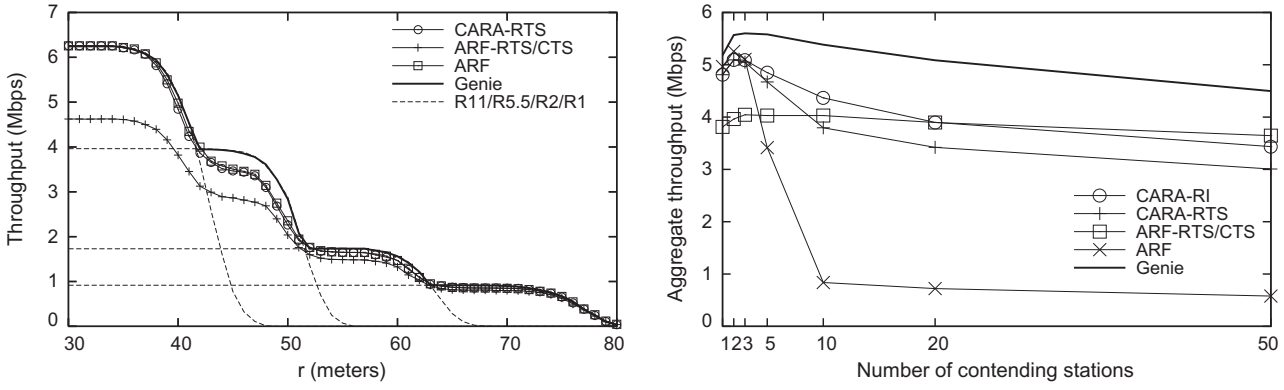
With a small number of contending stations, i.e., 1 or 2, ARF shows slightly better performance than CARA–RI and CARA–RTS due to the RTS/CTS overhead. However, the aggregate system throughput of ARF degrades drastically when the number of contending stations increases. For example, with more than 10 stations, the aggregate throughput with ARF drops to under 1 Mbps. There are two main reasons for ARF's ill behavior. First, as described in Section 4.3, ARF cannot differentiate collisions from channel errors. In consequence, a station may decrease its transmission rate over-aggressively and operate at lower rates. Second, since each contending station conducts its rate adaptation independently, they may end up transmitting at different rates. Such transmission rate diversity causes the following *performance anomaly* [28]: since the 802.11 DCF is designed to offer long-term equal medium access probabilities to all contending stations, the throughput of a high-rate station is always bounded below the lowest transmission rate of other stations in the network.

It is interesting to see that, when there are 50 contending stations, ARF–RTS/CTS shows slightly higher throughput than CARA–RI. As revealed in [20], it is desired that an RTS/CTS exchange always precedes a data frame transmission to reserve the channel when the collision is the dominant source of transmission failures. In addition to the effect of the number of stations, this can be more clearly observed with longer transmission duration based on which the usage of RTS/CTS becomes more effective [29]. Therefore, in this situation, ARF–RTS/CTS, which always enables the RTS/CTS exchange, performs better than CARA–RI which uses adaptive RTS/CTS. On the other hand, when the number of contending stations is small, ARF–RTS/CTS performs worse than CARA–RI due to the added RTS/CTS overhead.

The reason why CARA–RI outperforms CARA–RTS is as follows. When a transmission failure occurs, CARA–RTS resets the consecutive success count (m) immediately regardless of the actual cause of failure, while m with CARA–RI is reset only when the frame loss can be identified as a channel-error-caused failure. Therefore, CARA–RI adjusts more quickly to the improving wireless channel condition by increasing the transmission rate more proactively, and hence, is able to achieve higher aggregate throughput than CARA–RTS.

5.2.3. Hidden-station topology

We also evaluate the testing schemes with multiple hidden stations. The simulated hidden-station topology is set up as follows. Two groups of stations and the AP are placed along a line segment of 90 m with one group at each



(a) One-hop topology networks. (b) Star-topology networks.

Fig. 6. Throughput comparison in different network topologies with the AWGN wireless channel.

end and the AP in the middle of the segment. Stations belonging to the same group are placed very close to each other. The carrier sense range is set to the same as the maximum transmission range of 1 Mbps (i.e., 80 m), in order to guarantee that the AP can communicate with all the stations, while stations belonging to different groups are hidden to each other. In order to study the impact of hidden stations on the performances of testing schemes, we vary the number of contending stations from 2 to 20 (or equivalently, the group size from 1 to 10). Each station transmits continuously to the AP; thus collisions can often be observed at the AP. Each station selects its data frame size randomly (with 720 bytes on average). Simulation results are shown in Fig. 7, where each value is averaged over 10 simulation runs. We have the following observations.

First, ARF–RTS/CTS shows better throughput performance than both CARA–CCA and ARF, because the RTS/CTS exchange handles the hidden stations effectively. ARF suffers severe throughput degradation even with a small number of hidden stations in the network. This is because ARF misinterprets the cause of transmission failures, and hence, operates at 1 Mbps for most of the time. Consequently, data frames with prolonged transmission durations may in turn suffer even more collisions (as the collision probability becomes a function of the transmission duration in the presence of hidden stations), which results in very low throughput.

Second, CARA–RI yields the highest aggregate throughput among all testing schemes, which is close to the upper bound produced by Genie. This is because CARA–RI increases the transmission rate more proactively than all other schemes. As a result, data frames are generally transmitted at higher rates with CARA–RI; consequently, the time wasted due to collisions and retransmissions is shorter. For example, with four contending stations in the network, the average throughput improvement of CARA–RI over CARA–CCA is 87.8%. In fact, the collected transmission rate distributions with four contending stations show that, with CARA–RI, about 83.5% and 14.7% of the successfully-delivered date frames are transmitted at 5.5 Mbps and 11 Mbps, respectively, while CARA–CCA operates at 2 Mbps for most of the time.

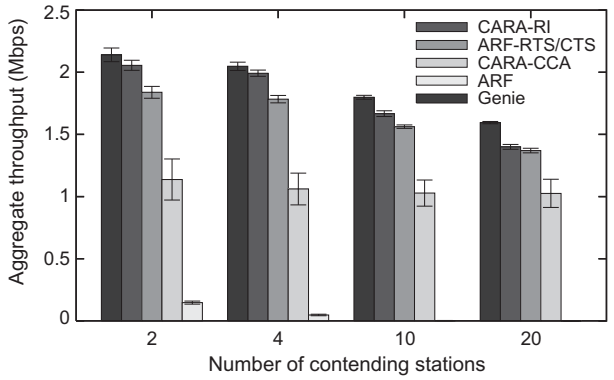


Fig. 7. Throughput comparison in hidden-station networks with the AWGN wireless channel and varying number of contending stations. 95% confidence intervals are shown as short line segments.

One may argue that the rationale to have CARA–RI's high throughput might be unbalanced throughput achievement among contending stations. To investigate this issue, we collect throughput achieved by each station and calculate Jain's fairness index.[4] We measure the fairness index of 0.98 as the minimum, which means that none of testing schemes raises fairness issue in this hidden topology.

5.2.4. Linear-chain topology

We expend the simulation environment to multi-hop communication, starting with linear-chain topologies. A chain topology is composed of seven stations and we label each station with N*x*; *x* indexes the set of involved stations, starting from N0. Each end station works as the source (N0) and destination (N6), respectively, which means that at the maximum six-hop data transfer can occur. The simulation environment changes by varying adjacent station distance, *r*. The results are averaged over 10 different random runs and are shown in Fig. 8a.

We observe that CARA schemes outperform ARF schemes in general. When $r \geqslant 21$, ARF shows a drastic throughput degradation compared with CARA schemes. It is because hidden stations arise when *r* increases more than or equal to 21 m; in the cases of $r \geqslant 21$, the hop count becomes more than two hops as shown in Fig. 8b. In order to mitigate the impact of hidden interference in such environments, the activation of RTS/CTS exchanges is inevitable for IEEE 802.11 MAC-based multi-hop communication. ARF–RTS/CTS however, does not illustrate high throughput compared with both CARA–RI and CARA–CCA. This indicates that the adaptive usage of RTS/CTS exchanges is more preferable than the always-enabled. In fact, ARF–RTS/CTS never shows higher throughput than CARA schemes for the entire distances.

Even in such multi-hop environments, CARA–RI outperforms CARA–CCA as observed in single-hop results. When $r = 21$ and 22, CARA–RI has 4.0% and 13.6% throughput enhancements compared with CARA–CCA, respectively. To identify detailed causes, we trace the rate selections of all considered schemes focusing on $r = 21$, 22, and 23. Fig. 9 shows the transmission rate distribution of the average number of successful transmissions at each station that is involved with data delivery; the *x*-axis represents the transmitting stations and the *y*-axis is the average number of successfully transmitted frames at each station. Each bar shows the distribution of used transmission rates and four bars at each transmitting station stand for 1: CARA–RI, 2: CARA–CCA, 3: ARF, and 4: ARF–RTS/CTS, from the left-hand side, respectively. Three different *r* values, i.e., $r = 21$, 22, and 23 are investigated.

We observe that the source station suffers from improper rate selections due to the transmission from existing hidden stations. In the case of $r = 21$ and 22, N4 becomes the hidden station with respect to the source station, N0. The successful frame reception at N2 is interfered by the transmissions from N4. Therefore, the source station em-

[4] Jain's fairness index (β) indicates the deviation of sample values and has the following implication: $0.5(\text{unfair}) \leqslant \beta = \frac{(\sum B_i)^2}{(n \sum B_i^2)} \leqslant 1(\text{fair})$, where B_i is each sample value and n is the number of samples.

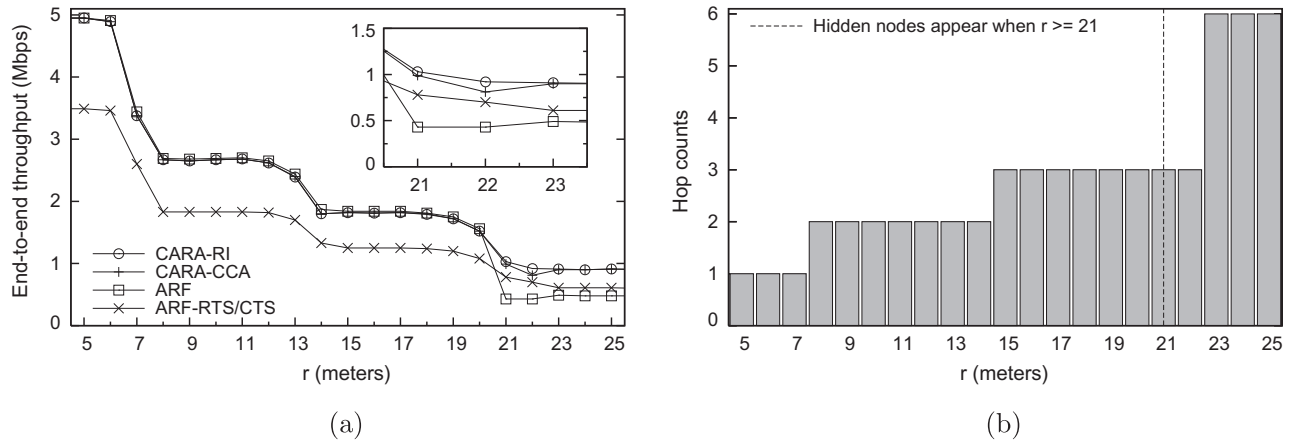


Fig. 8. Performance comparison in chain topologies with varying adjacent station distance (r): (a) throughput comparison of CARA schemes and ARF with/-out RTS/CTS and (b) corresponding hop counts that are identical for all schemes as the ETT-based optimal route selection is commonly used.

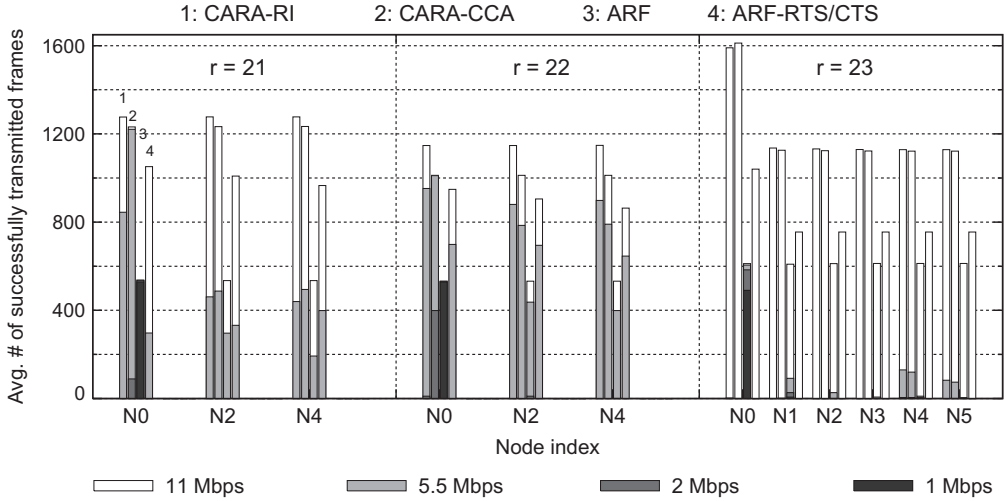


Fig. 9. Transmission rate distribution of the average number of successfully transmitted frames at each station; each bar shows the distribution of used rates and each scheme is enumerated as 1: CARA–RI, 2: CARA–CCA, 3: ARF, and 4: ARF–RTS/CTS.

ploys lower transmission rates than other transmitting stations in general, except the case of ARF–RTS/CTS. In the case of $r = 23$, the source station with CARA–CCA transfers the largest number of frames, even compared to that with CARA–RI. However, the delayed rate increase at N1 due to hidden interference makes CARA–CCA present less number of successes than CARA–RI eventually.

5.2.5. Square-grid topology with single source

We also investigate the effectiveness of CARA schemes in an extended topology, square-grid network. Fig. 10a illustrates the used 7×7 square-grid topology where 49 stations exist. All stations are candidate source stations except the GW (gateway) that is located at the right-upper corner. One out of 48 candidates is selected as a source and generates UDP traffic destined to the gateway. The ETT-based optimal route setup is done before generating a data flow: for example, N3 selects its routes as N3 $\mapsto$ N11 $\mapsto$ N19 $\mapsto$ N27 $\mapsto$ N34 $\mapsto$ N41, which is illustrated in Fig. 10a. The station separation distance is 20 m and hence, a 11 Mbps transmission is possible between adjacent stations.

We run all possible source cases with different transmission rate selection schemes, and the results are shown in Fig. 10b. The x-axis represents the throughput performance of CARA–RI and the y-axis is that of other schemes. Three different symbols differentiate the performances of CARA–CCA, ARF, and ARF–RTS/CTS as shown in the figure. The fact that a symbol is located under the $y = x$ line means that CARA–RI outperforms the compared scheme for the given source station. We observe that CARA–RI works better than ARF and ARF–RTS/CTS in most cases: due to the RTS/CTS overhead, CARA schemes can show minor throughput degradation compared with ARF. CARA–CCA shows quite comparable performance to that of CARA–RI in this environment.

5.2.6. Square-random topology with multiple sources

We now present the most generalized evaluation scenario: randomly located stations with multiple traffic sources. As illustrated in Fig. 11a, in total 49 stations are randomly located in a 120×120 m^2 area. To mark the basis locations, one of the candidate source stations, denoted

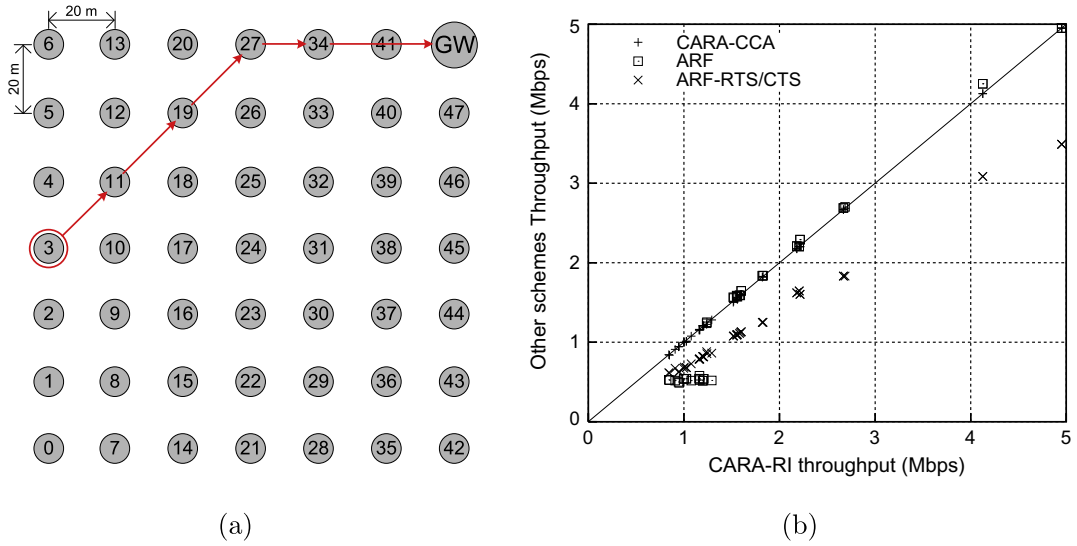


Fig. 10. (a) A 7×7 square-grid topology with the adjacent station distance of 20 m: for example, N3 chooses its route as $N3 \mapsto N11 \mapsto N19 \mapsto N27 \mapsto N34 \mapsto N41$ that results from the ETT-based optimal route calculation; (b) throughput comparison of CARA–RI and other rate selection schemes, i.e., CARA–CCA, ARF, and ARF–RTS/CTS, whose symbols are described in the legend, respectively.

as '0', is positioned at (100, 100) and the destination station ('48') has its position at (220, 220). The number of source stations varies from 1 to 32. For a given number of source station scenario, we generate 30 different combinations of source stations to cover almost every possible source combination, and the average results are shown in Fig. 11b.

The throughput of all four schemes decreases when the number of source stations increases up to two. Adding one more source station, the amount of collisions increases and hence, the aggregate throughput decreases. The reason why all schemes except ARF shows monotonically increasing throughput as the number of source stations increases is that the system-wise offered load with respect to the network increases.

Fig. 11b shows that CARA schemes outperform ARF schemes for the entire range of *x*-axis. As the offered load increases along with the number of source stations, more collisions happen, hence more struggles for each scheme to accurately select the transmission rate. We investigate the average cumulative number of successful transmissions for given rates with varying number of source stations. The relative portion of lower rate uses increases for the cases of CARA–CCA and ARF, as the number of source stations increases. CARA–RI and ARF–RTS/CTS show constant-rate usage irrespective of the number of source stations. Due only to the overhead of RTS/CTS, ARF–RTS/CTS shows degraded performance compared with CARA–RI.

5.3. Fading channel environments

In the second part of the simulation study (Sections 5.3.1 and 5.3.2), we simulate a Ricean fading channel [25]

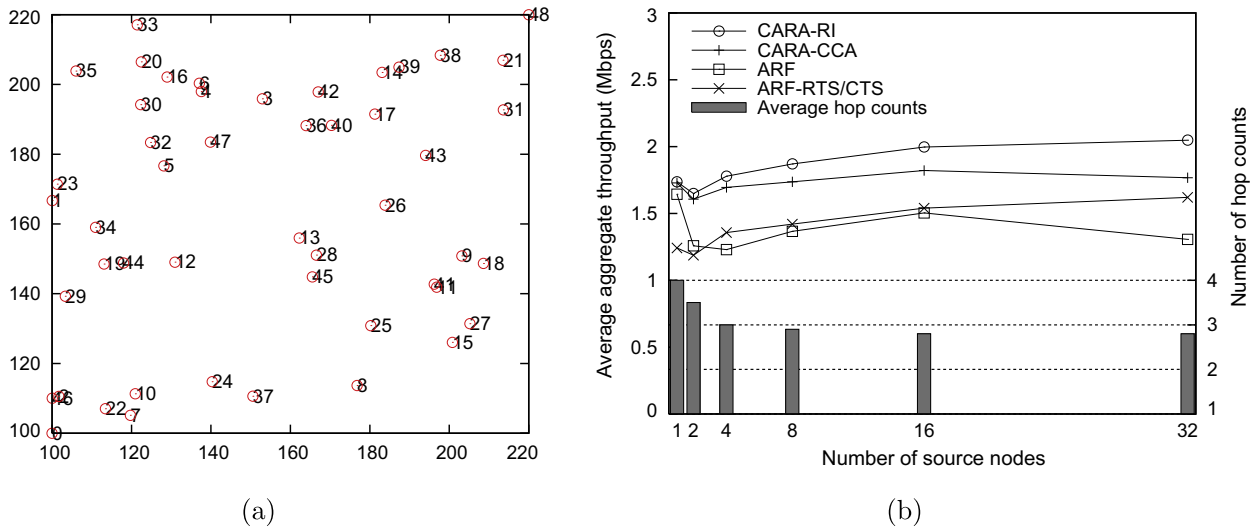


Fig. 11. (a) Distribution of randomly source stations: each station is indexed with a number at its right-hand side and (b) average aggregate throughput comparison of CARA–RI and other rate selection schemes.

with Ricean K factor of 3 dB to describe the indoor fading channel environment [30].

5.3.1. Station mobility

We first compare the testing schemes in the following simulation setup. A group of five closely-located stations are placed 20 m away from the AP. At time $t = 20$ s, the AP starts moving away from the stations along a straight line at the speed of 2 m/s; so at time $t = 45$ s, the AP is 70 m away from the stations. Fig. 12a shows the wireless channel variation between the AP and the stations in the interval $t = [25, 45]$ s by plotting the SNR values measured at the AP. Both short-term and long-term channel variations can be observed from the figure, which are caused by fading and station mobility, respectively. Shown as three dashed lines in Fig. 12a, the SNR values of 10.8 dB, 7.2 dB, and 3.7 dB are high enough to accommodate the transmission rate of 11 Mbps, 5.5 Mbps, and 2 Mbps, respectively under the simulation setup. Rate adaptation behaviors of different testing schemes are shown in Fig. 12b–f. Note that only the successful transmission attempts are shown in these figures. It can be seen clearly that, in terms of the capability of reacting to and adapting the transmission rate to the wireless channel variation, the testing schemes are ranked in the following order (from the best to the worst): Genie $\gg$ CARA − RI $\gg$ CARA − CCA $\gg$ CARA − RTS $\gg$ ARF. Consequently, CARA–RI yields the throughput of 1.94 Mbps during the time span, which is 13.5%, 28.5%, and 100.0% higher than that of CARA–CCA, CARA–RTS, and ARF, respectively.

To study in more detail how CARA–RI adapts the transmission rate to the wireless channel variation, we zoom in the interval $t = [27.5, 28.3]$ s, and plot the wireless channel variation against CARA–RI's rate selections in Fig. 13a. The traced transmission attempts are depicted using different symbols, standing for different meanings: a " +" point stands for a successful transmission, while " □" and " △" points represent transmission failures caused by channel errors and frame collisions, respectively; " ▽" points also represent frame collisions, but only those frame collisions that are detected by CCA Detection. Both collision-aware rate decrease and collision-aware rate increase can be observed clearly from the figure. Firstly, one can see that two consecutive transmission failures may not necessarily trigger rate decrease in CARA–RI. For example, two consecutive transmission attempts fail at approximately $t = 27.98$ s. However, since one of the two transmission failures (the one that is shown as a " ▽" point) is caused by frame collision and detected by CCA Detection, the transmission rate is not lowered. Secondly, one can see that CARA–RI increases the transmission rate in a more proactive way. For example, none of the first four collision

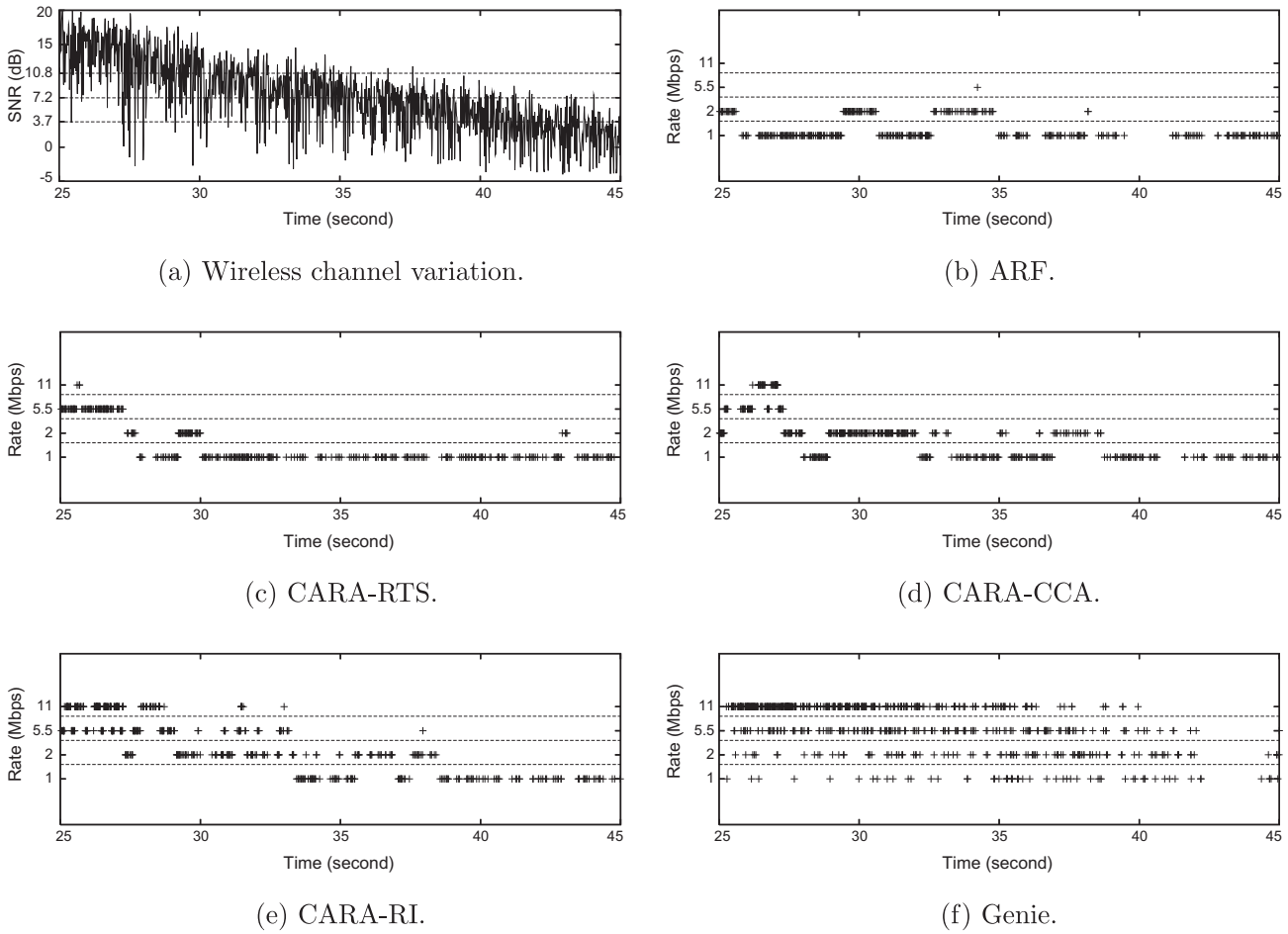


Fig. 12. Wireless channel variation and rate adaptation behaviors of testing schemes in a fading channel environment with station mobility.

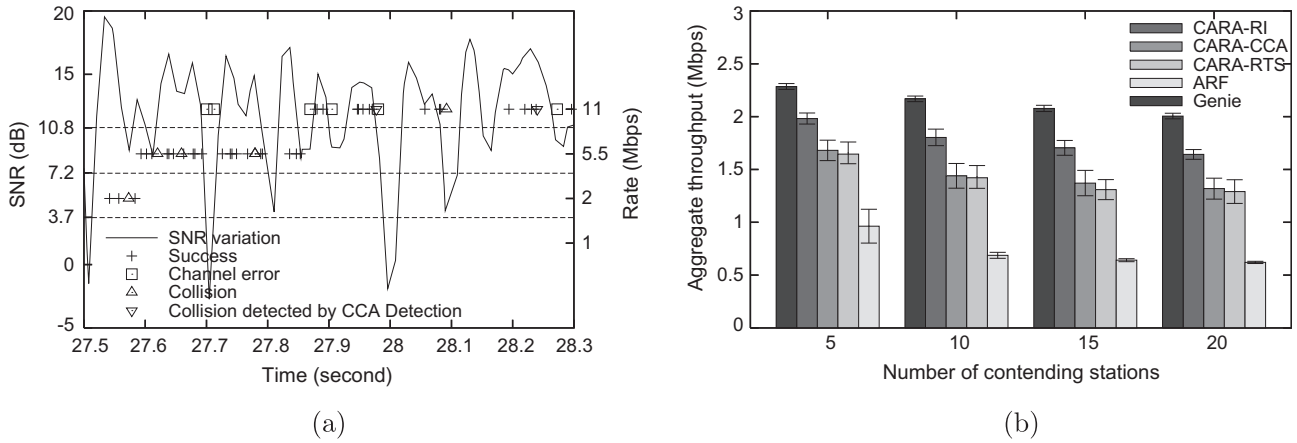


Fig. 13. (a) Rate adaptation behavior of CARA–RI and (b) throughput comparison in a fading channel environment with station mobility, where 95% confidence intervals are shown as short line segments.

events (i.e., the first four " △" points) in the figure delays the rate-increase decision; hence the transmission rate selections of CARA–RI follow the SNR variation pretty well.

We vary the number of stations in the group and simulation results are shown in Fig. 13b. Each value in the figure is obtained by averaging over 20 simulation runs. As expected from Fig. 12, CARA–RI yields higher throughput than both CARA–CCA and CARA–RTS, and outperforms ARF significantly, while CARA–CCA shows comparable throughput performance with CARA–RTS.

5.3.2. Static stations at random locations

Finally, we evaluate and compare the performances of testing schemes in randomly-generated network topologies: all stations are static and randomly placed within a circle around the AP with the radius of 80 m. The number of contending stations varies from 5 to 20. Simulation results are plotted in Fig. 14, where each value is averaged over 50 scenarios. Note that the simulation results for Genie are not shown in the figures. This is because the proper usage of RTS/CTS exchange in Genie requires an accurate estimation of the collision probability, which is nontrivial in the presence of randomly-located stations in the network.

In general, CARA schemes are significantly better than ARF in terms of aggregate throughput, thanks to their collision-awareness capabilities, and CARA–RI outperforms CARA–CCA. The performance comparison of testing schemes in terms of aggregate throughput becomes more clear in the scatter plot shown in Fig. 15. The left and right y-axes in each of four figures in Fig. 15 represent the CARA–RI throughput and the ARF throughput, respectively, and the CARA–CCA throughput is shown along the x-axis. "o" points represent the relation between CARA–RI and CARA–CCA throughput, and " +" points represent the relation between CARA–CCA and ARF throughput. In these figures, an "o" point above the $y = x$ line means that CARA–RI outperforms CARA–CCA, while a " +" point below the $y = x$ line means that CARA–CCA yields higher throughput than ARF. It can be seen clearly from Fig. 15 that CARA–RI outperforms CARA–CCA which yields much higher throughput than ARF in all the simulated scenarios.

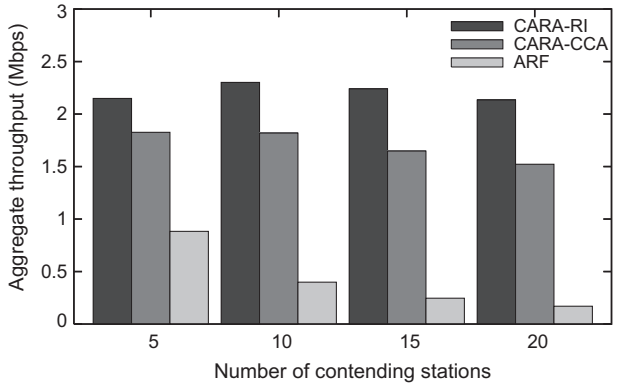


Fig. 14. Throughput comparison in random-topology networks with varying number of contending stations.

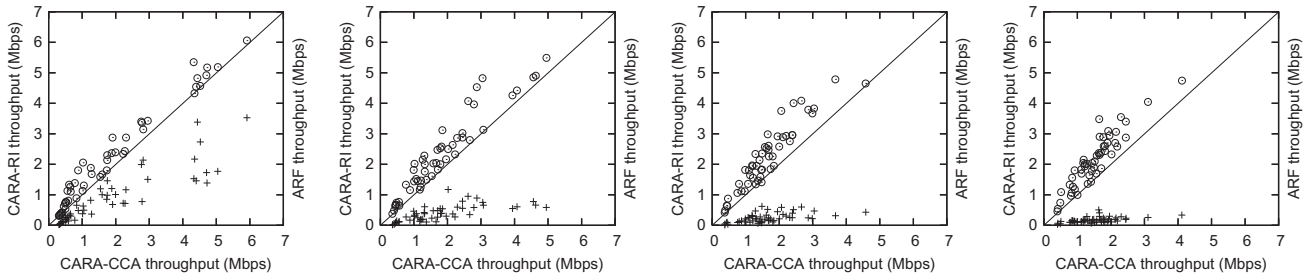
Note that there may be some hidden stations in random-topology networks. However, the chance of existence of hidden stations in randomly-generated network topologies is lower than in the hidden-station topology described in Section 5.2.3. Thus in our simulations, higher throughput is usually observed with random-topology networks than with hidden-node topology networks.

6. CARA implementation and experimentation

As a proof of concept, we implement CARA–RI, CARA–RTS, and ARF in our Linux-based WLAN testbed. In this section, we first describe the implementation details and the experimental setup, and then present evaluation results.

6.1. Implementation details

We compare CARA schemes with ARF (with and without RTS/CTS exchanges) and other available schemes in our experimental evaluation. We implement the functional operations of testing schemes based on MadWifi (version 0.9.2 ∼ 0.9.4) [4] which is an open-source Linux-based WLAN driver to control Atheros WLAN chipsets [31]. Lap-



(a) $n = 5$. (b) $n = 10$. (c) $n = 15$. (d) $n = 20$.

Fig. 15. Scatter plot of 50 random-topology network scenarios with different number of contending stations. "o" points represent the relation between CARA–RI and CARA–CCA throughput, and " +" points represent the relation between CARA–CCA and ARF throughput.

top and desktop computers in our testbed are loaded with Debian Linux (kernel version 2.6.18 ~ 2.6.24) [32] and the customized MadWifi driver to form a WLAN.

One core part of CARA is the adaptive usage of the RTS/CTS exchange. However, as pointed out in [5], most of the available Atheros WLAN chipsets are *high-latency* systems due to the high communication latency between their PHY and MAC layers. As a result, the per-MPDU-based RTS-on/off operation required by CARA schemes is quite challenging with the MadWifi driver. Specifically, there is the following operational challenge when implementing CARA schemes. All parameters related to a data frame transmission are specified in the form of a descriptor. Once a data frame is ready to be transmitted and is passed to a buffer in the driver, namely *ath_txq*, all operations regarding the data frame transmission should follow the rule specified in the descriptor. No operational control change however is allowed for data frames backlogged in *ath_txq*. The RTS-on operation in CARA schemes, which should be enabled upon the detection of a data transmission failure, cannot be invoked in a per-MPDU manner after getting into the driver buffer.

Our driver patch to cope with this challenge is to limit the maximum number of data frames that can be queued in *ath_txq* to 1. We have worked on analyzing the effect of the queue-restriction patch on the system performance, and concluded that our approach is not harmful in achieving the identical maximum throughput to that without the queue restriction.

6.2. Experimental setup

We construct a Linux-based WLAN testbed to evaluate the implemented schemes. Cisco 802.11a/b/g WLAN adapters [33], which embed Atheros AR5002X chipsets, are used in our testbed, and our modified MadWifi driver is loaded into the testbed computers. In our experiments, both 802.11a/g configurations are used. Meanwhile, the results based on the 802.11a setting are mainly presented in this section because of the observed consistent tendency, which we believe comes from 802.11a's cleaner frequency band, thus yielding less interference from the surrounding environment. All experiments are performed in an indoor laboratory.

We use Iperf [34] as the UDP packet generator and it is loaded into both the AP and stations. CBR (Constant Bit Rate) traffic is generated at a certain rate to make sure that each station always has a frame *to* transmit and *a*th_*t*xq is never empty. CBR packet size is 1000 octets, unless specified otherwise. Our empirical investigation of rate selection performance includes three collision-ambient scenarios: (1) static location with contention, (2) station mobility with contention, and (3) static location, but enhanced contention from a hidden station. Detailed topological configurations will be addressed *at each scenario.*

6.3. Static location with contention

As addressed earlier, our goal in this experiments is to show that the real testbed results confirm what we observed in simulation studies. The first scenario operates seven WLAN entities (i.e., six stations and one AP), which targets to double-check the advantage of collision-aware rate selection when a hidden station does not exist.

Stations are located very close to each other enough not to produce the hidden-station effect. The AP contends for the medium while it is approximately 6.8 m away from the stations. In our experimental configuration, it is highly likely that the channel quality over the distance is good enough to afford error-free communication among the WLAN entities at the fastest rate, i.e., 54 Mbps. Each experimental run lasts for 10 min, and the results for each scenario are averaged over 12 runs. To build a highly contending environment even with a small number of stations (up to six), we make the contention window size remain the minimum value. Furthermore, in order to minimize potential unexpected performance variation caused by people's movement and interference from other devices operating at the same frequency band, experiments are conducted at nighttime and all WLAN adapters are configured to operate at channel 157 of the 802.11a PHY (with the center frequency of 5.785 GHz), which is found to be the cleanest channel during our experimental study.

Fig. 16 plots the measured performance in terms of MAC throughput with 95% confidence interval. As observed from the simulation study in Section 5, Fig. 16 shows that CARA–RI outperforms both CARA–RTS and ARF, and CARA–RTS yields higher throughput than ARF, particularly when

the number of contending stations (denoted as N for the rest of this section) is large. For example, when $N = 6$, CARA–RI and CARA–RTS show 305.4% and 272.2% throughput improvement over ARF, respectively. On the other hand, when N is small, all three schemes show comparable throughput performances due to less contention and collisions in the network. On average, the throughput improvements of CARA–RI and CARA–RTS over ARF are about 62.0% and 57.5%, respectively.

In order to have a good understanding on how/why CARA schemes outperform ARF, we investigate the cause of observed throughput differences by plotting the ECDF (Empirical Cumulative Distribution Function) of the rates used for successful data frame transmissions with ARF, CARA–RTS, and CARA–RI in Fig. 17a–c, respectively. Each curve in the figure intersects with the right y-axis at value p – which represents the percentage of successful transmissions out of total number of transmission attempts – when the corresponding testing scheme is used. Clearly, p is less than or equal to one, and varies with N for a given testing scheme. In general, as N increases, there are more contention and collisions in the network, thus resulting in a smaller p.

As shown in Fig. 17a, when there is only a single transmitter in the network (i.e., $N = 1$), almost all the data frames are transmitted at the highest 54 Mbps, and almost all transmissions succeed, meaning that the channel condition is good enough to allow highest-rate data transmissions. However, with ARF, when there are more contending stations in the network, there are more transmission failures and majority of successful data transmissions are conducted at lower rates. For example, when $N = 6$, about 30% of data frame transmissions fail, and about 60% of data frames are transmitted successfully but at the lowest 6 Mbps. Such an undesirable transmission rate distribution clearly testifies to ARF's conservative nature as well as its inability to differentiate frame collisions from transmission failures caused by deteriorated channel conditions. This is consistent with our observations in Fig. 16.

On the other hand, thanks to collision-aware rate-decrease decisions in CARA–RTS, data frames are transmitted at relatively higher rates in contending network environments. As shown in Fig. 17b, with CARA–RTS, most successful data frame transmissions are conducted at 36, 48, or 54 Mbps even when $N = 6$. In Fig. 17c, we observe even better transmission rate distributions with CARA–RI, which are nearly the same regardless of the number of contending stations in the network. This is due to CARA–RI's collision-aware rate-increase decisions in addition to collision-aware rate-decrease decisions as in CARA–RTS. Accordingly, as shown in Fig. 16, CARA–RI shows slightly higher aggregate throughput than CARA–RTS; the limited performance improvement is due to the fact that the throughput improvement of transmitting at 54 Mbps over transmitting at 48 or 36 Mbps is not significant.

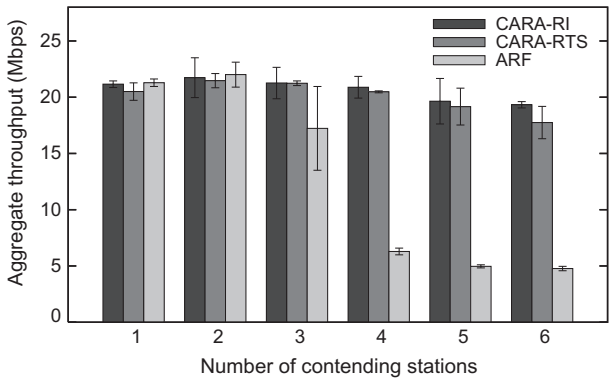


Fig. 16. Throughput comparison of CARA–RI, CARA–RTS, and ARF with varying number of contending stations in our Linux-based WLAN testbed, where 95% confidence intervals are shown as short line segments.

6.4. Station mobility

In the second scenario, we revisit the impact of station mobility on transmission rate selection, which we observed previously in Section 5.3.1 with simulation studies.

There is a total of 10 computers used in the experiments: one laptop for AP and nine desktop computers for stations. At the initial stage, a group of closely-located stations are placed inside a room, which are about 4 m away from the AP that is in front of the door. After starting each experiment, the AP begins moving away from the stations along a corridor at the speed of 0.54 m/s approximately. At 30 s, the AP reaches the end of the corridor and comes back to the door at the same speed. At the time of 60 s, an experiment finishes and the AP returns to the initial location.

We measure the throughput of each station and used transmission rates. The average aggregate throughput performance with 95% confidence interval is shown in Fig. 18. In general, the aggregate throughput degrades as the number of contending stations increases. Rate adaptation behaviors of different testing schemes are shown in Fig. 19a–c. Note that Fig. 19 depicts the used transmission rate, irrespective of its success or failure. In terms of the capability of reacting to the wireless channel fluctuation and adapting the transmission rate, the testing schemes are ranked in the order of CARA–RI $\gg$ CARA–RTS $\gg$ ARF (from the best to the worst), which confirms what we observed using simulation studies in Section 5.3.1.

The rate distribution of ARF–RTS/CTS is particularly interesting. As shown in Fig. 19b, its distribution does not vary much with the number of contending stations (N) from 3 to 9, due to its per-frame RTS activation. As a result, the standard deviation of the measured throughput of ARF–RTS/CTS for different N is only 0.87 Mbps, while all other schemes show a deviation between 2.24 and 2.84 Mbps. Accordingly, the rate distribution of ARF–RTS/CTS can be the reference that a proper rate adaptation scheme should follow under this experimental setup. With $N = 3$, CARA schemes outperform ARF–RTS/CTS due to the intelligent RTS activation. As the number of stations increases, however, CARA–RTS shows huge throughput degradation. This is because intense transmission error sources such as long-term and short-term channel variation as well as collision induced by contention generate transmission errors that make m be reset, thus yielding improper (or delayed) rate-increase attempts. In comparison,

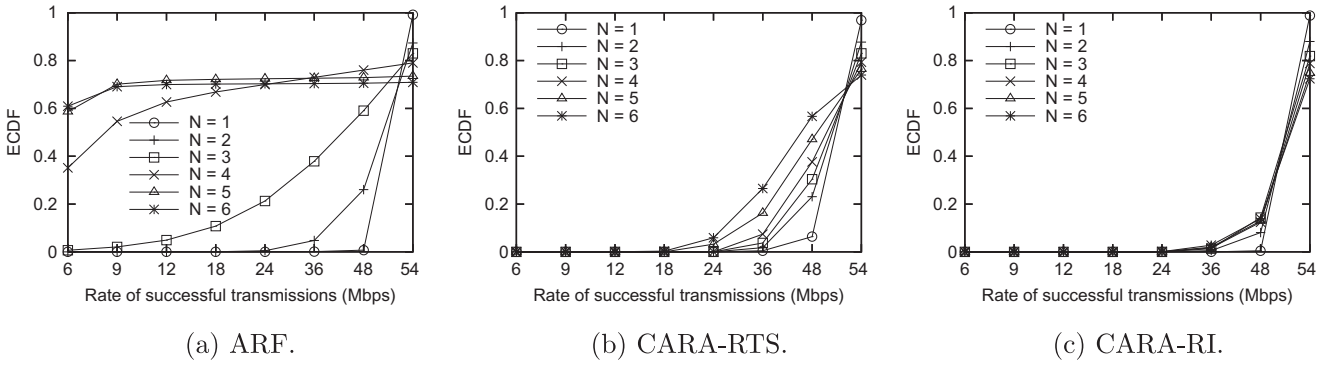


Fig. 17. ECDF of the rates of successful transmissions for ARF, CARA–RTS, and CARA–RI.

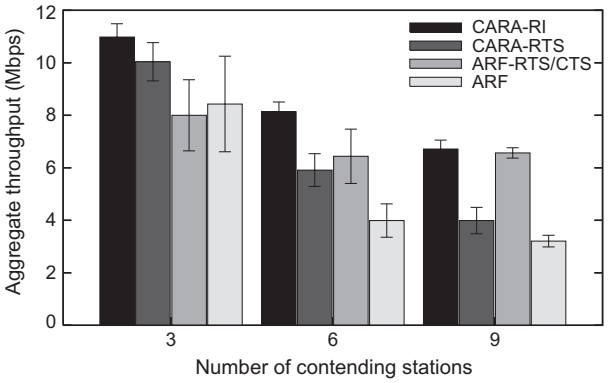


Fig. 18. Throughput comparison of CARA–RI, CARA–RTS, ARF–RTS/CTS, and ARF with varying number of contending stations in our Linux-based WLAN testbed, where 95% confidence intervals are shown as short line segments.

thanks to CARA–RI's RTS activation before resetting m, CARA–RI shows a comparable rate distribution with ARF–RTS/CTS when $N = 9$, which contributes to the highest throughput yielded by CARA–RI among all testing schemes.

6.5. Static location with hidden stations

The *third* scenario deals with hidden-station-induced collisions and their impact on rate adaptation schemes. To generate a hidden station topology, we place a receiver in between two transmitters. To verify that these transmitters are hidden to each other, we generate broadcast traffic at both the transmitters. The number of broadcast frames transmitted by each station remains nearly the same, irrespective of the presence or absence of the other transmitter. This suffices that the transmitters are mutually hidden. We refer to one as *transmitter* and the other as *hidden interferer*. The hidden interferer generates 1003-byte UDP broadcast frames at different traffic generation rates to expose the transmissions from the transmitter to the receiver to mild, moderate, and intense interference. The transmitter also generates the same size of UDP frames to the receiver at the generation rate enough to operate in a saturated mode. Each experiment run lasts for 90 s and results are averaged over three runs for each of the evaluated schemes.

We have compared CARA schemes with ARF. To evaluate CARA one step further, we now consider other existing schemes instead of ARF: SAMPLE (SampleRate) [10], ONOE [4], and AMRR (Adaptive Multi-Rate Retry) [5]. These schemes are well-known in the MadWifi community and operate as follows:

SAMPLE estimates the transmission time associated with each transmission rate under the influence of frame losses. Based on the estimation, it generates a statistical table, updates it with each frame transmission, and adapts its transmission rate by looking it up. SAMPLE performs a rate upgrade if either 2 s have elapsed operating at the current rate or the estimated transmission time of the current rate is greater than or equal to twice the estimated transmission time of the best rate. On the other hand, SAMPLE downgrades its transmission rate with four consecutive

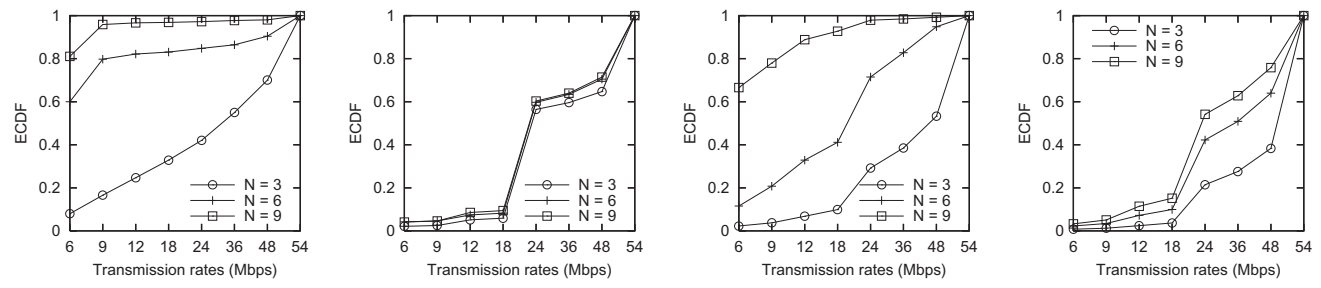


Fig. 19. ECDF of the used transmission rates for ARF, ARF–RTS/CTS, CARA–RTS, and CARA–RI.

Table 2
Throughput comparison of CARA–RTS, ONOE, SAMPLE, and AMRR with varying traffic generation rate of interferer. The gain is calculated over the average throughput of other three schemes.

Interferer rate (Mbps)	CARA–RTS	ONOE	SAMPLE	AMRR	Gain over avg. (%)
0	15.50	14.40	14.10	13.30	11.2
5	10.01	9.46	9.28	6.40	19.5
10	3.60	0.63	0.66	0.62	464.3
15	3.01	0.59	0.60	0.50	434.0
20	2.09	0.46	0.49	0.40	364.4
25	1.11	0.27	0.29	0.17	356.2

failures at the current rate. ONOE tags each transmission rate with a credit. Credit count increases by unity (up to the maximum, 10) with less than 10% of failures out of the total transmissions. A node executes the above rate adjustment policy (i.e., credit update) every second. When a credit associated with a rate reaches ten, the transmission rate increases, which means at least 10 s is taken for a rate-up. ONOE lowers the transmission rate after four failures in a raw and then makes it the lowest with other two consecutive failures. AMRR sustains a transmission rate with less than 33% frame losses. If the frame losses are less than 10%, it increases the rate. For consecutive frame losses, AMRR decreases the transmission rate one-step down for every failure and fixes the rate to the lowest with its fourth attempt. It is intriguing that AMRR does not do a retransmission after the fourth attempt.

Table 2 shows the throughput performances of the compared schemes. The maximum achievable throughput of different schemes are measured when the hidden interferer varies its offered load. SAMPLE, ONOE, and AMRR perform poorly with significant throughput loss as the interference intensity shifts from mild (up to 5 Mbps) to intense (from 10 Mbps) from the hidden interferer. CARA–RTS outperforms all compared schemes for the entire range of interference. The gain of CARA–RTS over the average throughput of other compared schemes reaches 464.3% (the 10 Mbps case). CARA–RTS achieves 1.11 Mbps even with the most severe interfering environment. This result shows that CARA effectively copes with such a pessimistic hidden environment, thus yielding a high-throughput gain.

7. Concluding remarks

In this paper, we present the design and implementation details of a novel collision-aware rate adaptation scheme called CARA for IEEE 802.11 WLANs. The key idea of CARA is that the transmitter combines adaptively the RTS/CTS exchange with the CCA functionality in order to differentiate frame collisions from frame transmission failures caused by channel errors. CARA specifies three different methods: (i) CARA–RTS that identifies collision via RTS Probing; (ii) CARA–CCA that identifies collision via RTS Probing as well as CCA Detection; and (iii) CARA–RI that makes collision-aware rate-increase decisions while identifying collision via CARA–CCA.

Through extensive ns-2 simulation study, we show that CARA schemes are more likely to make correct rate adaptation decisions than other schemes in various environmental settings. Furthermore, compared with CARA–RTS and CARA–CCA, CARA–RI adjusts more quickly to the improving wireless channel condition by increasing the transmission rate more proactively, thus showing better throughput performance in various network conditions. In addition to the simulation study, we implement CARA schemes on the MadWifi platform in our Linux-based WLAN testbed in order to prove the superiority of CARA to existing schemes, such as ARF, *ARF–RTS/CTS,* SAMPLE, ONOE, and AMRR. Experimental results also demonstrate the effectiveness of CARA.

Acknowledgement

The authors thank Okhwan Lee and Yeonchul Shin for their participation in the experiments. This work was supported by the IT R&D program of MKE/KEIT.

References

[1] A. Kamerman, L. Monteban, WaveLAN-II: a high-performance wireless LAN for the unlicensed band, Bell Labs Technical Journal 2 (3) (1997) 118–133.
[2] A.J. van der Vegt, Auto Rate Fallback Algorithm for the IEEE 802.11a Standard.
[3] P. Chevillat, J. Jelitto, A.N. Barreto, H.L. Truong, A dynamic link adaptation algorithm for IEEE 802.11a wireless LANs, in: Proceedings of IEEE ICC'03, Anchorage, AK, USA, 2003, pp. 1141–1145.
[4] MadWiFi: Multiband Atheros Driver for WiFi. <http://madwifi.org/>.
[5] M. Lacage, M.H. Manshaei, T. Turletti, IEEE 802.11 rate adaptation: a practical approach, in: Proceedings of ACM MSWiM'04, Venezia, Italy, 2004.
[6] D. Qiao, S. Choi, Fast-responsive link adaptation for IEEE 802.11 WLANs, in: Proceedings of IEEE ICC'05, Seoul, Korea, 2005, pp. 3583–3588.
[7] D. Qiao, S. Choi, Goodput enhancement of IEEE 802.11a wireless LAN via link adaptation, in: Proceedings of IEEE ICC'01, Helsinki, Finland, 2001, pp. 1995–2000.
[8] D. Qiao, S. Choi, K.G. Shin, Goodput analysis and link adaptation for IEEE 802.11a wireless LANs, IEEE Transactions on Mobile Computing 1 (4) (2002) 278–292.
[9] J. del Prado Pavon, S. Choi, Link Adaptation strategy for IEEE 802.11 WLAN via received signal strength measurement, in: Proceedings of IEEE ICC'03, Anchorage, AK, USA, 2003, pp. 1108–1113.
[10] J. Bicket, Bit-rate Selection in Wireless Networks, Master's Thesis, MIT, 2005.
[11] S.H.Y. Wong, H. Yang, S. Lu, V. Bharghavan, Robust Rate adaptation for 802.11 wireless networks, in: Proceedings ACM MobiCom'06, Los Angeles, CA, USA, 2006, pp. 146–157.
[12] X. Chen, D. Qiao, J. Yu, S. Choi, Probabilistic-based rate adaptation for IEEE 802.11 WLANs, in: Proc. IEEE GLOBECOM'07, Washington, DC, USA, 2007, pp. 4904–4908.
[13] J. Zhang, K. Tan, J. Zhao, H. Wu, Y. Zhang, A practical SNR-guided rate adaptation, in: Proceedings of IEEE INFOCOM'08, Las Vegas, Nevada, USA, 2008, pp. 2083–2091.

[14] G. Holland, N. Vaidya, P. Bahl, A rate-adaptive MAC protocol for multi-hop wireless networks, in: Proceedings of ACM MobiCom'01, Rome, Italy, 2001, pp. 236–251.
[15] B. Sadeghi, V. Kanodia, A. Sabharwal, E. Knightly, Opportunistic media access for multirate Ad Hoc networks, in: Proceedings of ACM MobiCom'02, Atlanta, GA, USA, 2002, pp. 24–35.
[16] J. Kim, S. Kim, S. Choi, D. Qiao, CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs, in: Proceedings of IEEE INFOCOM'06, Barcelona, Spain, 2006.
[17] S. Kim, S. Choi, D. Qiao, J. Kim, Enhanced rate adaptation schemes with collision awareness, in: Proceedings of IFIP/TC6 Networking'07, Atlanta, GA, USA, 2007.
[18] S. Choi, K. Park, C.-K. Kim, On the performance characteristics of WLANs: revisited, in: Proceedings of ACM Sigmetrics'05, Banff, Alberta, Canada, 2005, pp. 97–108.
[19] IEEE Std., IEEE 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, August 1999.
[20] G. Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on Selected Areas in Communications 18 (3) (2000) 535–547.
[21] P.E. Bausbacher, J.L. Kearns, Transmission parameter selection in an adaptive packet-radio network, in: Proceedings of Tactical Communications Conference'90, Fort Wayne, IN, USA, 1990, pp. 51–68.
[22] The Network Simulator – ns-2. <http://www.isi.edu/nsnam/ns/>.
[23] Intersil, HFA3861B; Direct Sequence Spread Spectrum Baseband Processor, January 2000.
[24] T.S. Rappaport, Wireless Communications: Principle and Practice, second ed., Prentice-Hall, 2002.
[25] R.J. Punnoose, P.V. Nikitin, D.D. Stancil, Efficient simulation of Ricean fading within a packet simulator, in: Proceedings of IEEE VTC'00, Boston, MA, USA, 2000, pp. 764–767.
[26] R. Draves, J. Padhye, B. Zill, Routing in multi-radio, multi-hop wireless mesh networks, in: Proceedings of ACM MobiCom'04, Philadelphia, PA, USA, 2004, pp. 114–128.
[27] D.S.J. De Couto, D. Auayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless networks, in: Proceedings of ACM MobiCom'03, San Diego, CA, USA, 2003, pp. 134–146.
[28] M. Heusse, F. Rousseu, G. Berger-Sabbatel, A. Duda, Performance anomaly of 802.11b, in: Proceedings of IEEE INFOCOM'03, San Francisco, CA, USA, 2003, pp. 836–843.
[29] I. Tinnirello, S. Choi, Y. Kim, Revisit of RTS/CTS exchange in high-speed IEEE 802.11 networks, in: Proceedings of IEEE WoWMoM'05, Taormina, Italy, 2005.
[30] M. Carroll, T.A. Wysocki, Fading characteristics for indoor wireless channels at 5 GHz unlicensed bands, in: Proceedings of Sympotic'03, Bratislava, Slovakia, 2003, pp. 102–105.
[31] Atheros Communications. <http://www.atheros.com/>.
[32] Ubuntu: A Community Developed, Linux-based Operating System. <http://www.ubuntu.com/>.
[33] Cisco Systems. <http://www.cisco.com/en/US/products/hw/wireless/>.
[34] Iperf: The TCP/UDP Bandwidth Measurement Tool. <http://dast.nlanr.net/Projects/Iperf/>.

Seongkwan Kim is a Senior Engineer at Samsung Electronics Co., Ltd., where he is currently working on the design of 4G telecommunications system such as IEEE 802.16e/m Mobile WiMAX. He received his Ph.D. degree in Electrical Engineering and Computer Science from Seoul National University in 2009, and received his BS and MS degrees in electronics engineering from Korea University in 2002 and 2004, respectively. He was with Hewlett–Packard Laboratories, Palo Alto, CA, USA as a visiting scholar in 2006. His research interests include algorithmic design and protocol development for various practical communication systems such as IEEE 802.11/15/16, cognitive radio, sensor/ad hoc network, wireless mesh network. He is a winner of the Best Paper Award in the 28th Korean Institute of Information Scientists and Engineers Student Paper Contest, and the Grand Prize in the 3rd IEEE Student Paper Contest awarded by IEEE Seoul Section. He was a recipient of IEEE Technical Committee on Computer Communications (TCCC) Student Travel Grant Award. He also received a Bronze Prize in the 12th Samsung Humantech Thesis Prize. He is a member of the IEEE.

Lochan Verma received the MS degree in Electrical Engineering and Computer Science from Seoul National University, South Korea in 2008 and bachelors degree in Information Technology from the National Institute of Technology, Karanataka, India in 2004. Prior to joining Seoul National University he worked as a senior software engineer with Samsung India Software Operations, Bangalore, India till 2006 and since 2008 is working as a research engineer with Samsung Electronics, Suwon, South Korea. He is affiliated with the Media Processing Lab. in Digital Media and Communication R&D Center at Samsung Electronics and is an active member of WFA (Wi-Fi Alliance) and a passive observer of WGA (Wireless Gigabit Alliance) industry alliances. He has 6 patents in filing and his research interests include wireless networks with emphasis on wireless LAN/MAN/PAN, mesh networks, L2/L3 (Layer) protocols, cross-layer approaches, multi-band operation in TV white spaces, 2.4, 5, and 60 GHz spectrum, Wi-Fi Direct, and Wi-Fi Display.

Sunghyun Choi is currently a visiting associate professor at the Electrical Engineering Department, Stanford University, USA, and an associate professor at the School of Electrical Engineering, Seoul National University (SNU), Seoul, Korea. Before joining SNU in September 2002, he was with Philips Research USA, Briarcliff Manor, New York, USA as a Senior Member Research Staff and a project leader for three years. He received his BS (summa cum laude) and MS degrees in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 1992 and 1994, respectively, and received Ph.D. at the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor in September, 1999. His current research interests are in the area of wireless/mobile networks with emphasis on wireless LAN/MAN/PAN, next-generation mobile networks, mesh networks, cognitive radios, resource management, data link layer protocols, and cross-layer approaches. He authored/coauthored over 140 technical papers and book chapters in the areas of wireless/mobile networks and communications. He has co-authored (with B. G. Lee) a book "Broadband Wireless Access and Local Networks: Mobile WiMAX and WiFi," Artech House, 2008. He holds 19 US patents, 10 European patents, and 11 Korea patents, and has tens of patents pending. He has served as a General Co-Chair of COMSWARE 2008, and a Technical Program Committee Co-Chair of ACM Multimedia 2007, IEEE WoWMoM 2007 and IEEE/Create-Net COMSWARE 2007. He was a Co-Chair of Cross-Layer Designs and Protocols Symposium in IWCMC 2006, 2007, and 2008, the workshop co-chair of WILLOPAN 2006, the General Chair of ACM WMASH 2005, and a Technical Program Co-Chair for ACM WMASH 2004. He has also served on program and organization committees of numerous leading wireless and networking conferences including IEEE INFOCOM, IEEE SECON, IEEE MASS, and IEEE WoWMoM. He is also serving on the editorial boards of IEEE Transactions on Mobile Computing, IEEE Wireless Communications, ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), Journal of Communications and Networks (JCN), Computer Networks, and Computer Communications. He has served as a guest editor for IEEE Journal on Selected Areas in Communications (JSAC), IEEE Wireless Communications, Pervasive and Mobile Computing (PMC), ACM Wireless Networks (WINET), Wireless Personal Communications (WPC), and Wireless Communications and Mobile Computing (WCMC). He gave a tutorial on IEEE 802.11 in ACM MobiCom 2004 and IEEE ICC 2005. From 2000 to 2007, he was a voting member of IEEE 802.11 WLAN Working Group. He has received a number of awards including the Young Scientist Award awarded by the President of Korea (2008); IEEK/IEEE Joint Award for Young IT Engineer (2007); the Outstanding Research Award (2008) and the Best Teaching Award (2006) both from the College of Engineering, Seoul National University; the Best Paper Award from IEEE WoWMoM 2008; and Recognition of Service Award (2005, 2007) from ACM. Dr. Choi was a recipient of the Korea Foundation for Advanced Studies (KFAS) Scholarship and the Korean Government Overseas Scholarship during 1997–1999 and 1994–1997, respectively. He is a senior member of IEEE, and a member of ACM, KICS, IEEK, KIISE.

Daji Qiao is currently an Associate Professor in the Department of Electrical and Computer Engineering, Iowa State University, Ames, Iowa. He received his Ph.D. degree in Electrical Engineering: Systems from The University of Michigan, Ann Arbor, Michigan, in February 2004. His current research interests include system modeling, performance analysis, algorithm innovation and implementation for various types of wireless/mobile networks, including WLAN, WiMAX, mesh networks and sensor networks. He is a member of IEEE and ACM.