A machine learning approach for dynamic control of RTS/CTS in WLANs

Yalda Edalat
Department of Computer Engineering, University of California Santa Cruz, USA
yalda@soe.ucsc.edu

Katia Obraczka
Department of Computer Engineering, University of California Santa Cruz, USA
katia@soe.ucsc.edu

Bahador Amiri
Department of Electrical Engineering, University of California Santa Cruz
Quantenna Communications
San Jose, USA
bamiri@soe.ucsc.edu

ABSTRACT

In this paper, we proposed a novel algorithm to dynamically enable and disable IEEE 802.11 DCF's RTS/CTS handshake. We start by conducting an experimental characterization of the performance of RTS/CTS as a function of packet size, transmission rate, and network contention, which complements existing work that evaluated RTS/CTS performance analytically and empirically. Motivated by our experimental evaluation of RTS/CTS performance, our algorithm uses current packet size and transmission rate, as well as an estimate of network contention to dynamically decide whether to use RTS/CTS or not. To the best of our knowledge, the proposed algorithm is the first to enable and disable the RTS/CTS handshake based on a set of current network conditions, and automatically adapt as these conditions change. Simulation results using a variety of WLAN scenarios, including synthetic and real traffic traces, demonstrate that the proposed approach consistently outperforms current best practices, such as never enabling RTS/CTS or setting the *RTS Threshold* (*RT*), which is used to decide whether to switch RTS/CTS on or off, to a static value.

CCS CONCEPTS

• **Networks → Wireless local area networks**; *Link-layer protocols*;

KEYWORDS

IEEE 802.11, WLAN, enabling/disabling RTS/CTS, machine learning

1 INTRODUCTION

The IEEE 802.11 standard, also known as WiFi, specifies both physical layer (PHY) and medium-access control (MAC) protocols for wireless local area network (WLAN) communication [1]. It is considered the de-facto standard for WLANs and, as such, has attracted considerable attention from both networking researchers and practitioners over the years. The first IEEE 802.11 standard was released in 1997, and since then, has grown to a large family of WLAN protocols as new frequency bands and PHY technologies became available. IEEE 802.11 defines two different types of MAC protocols, a mandatory one called Distributed Coordination Function (DCF) and an optional one called Point Coordination Function (PCF), built atop of DCF. IEEE 802.11 DCF, which is by far more widely deployed than its PCF counterpart, arbitrates access to the shared communication medium using a random-access (or contention-based) approach, in particular the Carrier Sense Multiple Access (CSMA) [17] protocol with or without collision avoidance (CA) [2]. In other words, IEEE 802.11 DCF defines a *base mode* which uses physical carrier sensing and a link-layer acknowledgment (ACK) to confirm correct reception of the transmitted data frame. DCF also specifies an optional mode which employs both physical- (i.e., CSMA) as well as virtual (i.e., CA) carrier sensing. As described in more detail in Section 2, CSMA/CA [2] was proposed as a way to combat the so-called *hidden terminal* problem. It allows nodes to reserve the channel before engaging in data communication by exchanging short control frames, namely Request to Send (RTS) and Clear to Send (CTS) ahead of transmitting data. RTS/CTS has been part of the IEEE 802.11 standard since its early versions and has been in use since then including more recent variants such as 802.11n and 802.11ac. However, as described in Section 2, the RTS/CTS handshake can also negatively impact performance since it introduces additional delay and overhead.

For this reason, IEEE 802.11 has defined a configurable parameter named *RTS Threshold (RT)*, which is used to enable and disable the RTS/CTS exchange. However, the standard does not specify what *RT* value(s) to use. For example, in some implementations, *RT* is set such that for small data frames, DCF's base mode is used. Otherwise, RTS/CTS is used when frame size is large enough. There are also cases where the *RT* value is set to the maximum frame size, so that RTS/CTS is never used. However, 802.11 product manufacturers make recommendations to users that if they are experiencing degraded performance, they should test their network with lower *RT* values.

As described in more details in Section 3, a number of studies have explored techniques to dynamically set the value of the *RT* based not only on packet size but also on other characteristics (e.g., transmission rate) and conditions (e.g., packet delivery ratio).

In this paper, we start by conducting an empirical characterization of RTS/CTS performance as a function of a number of factors including packet size, transmission rate (for both data and control), and network contention. Our experimental RTS/CTS performance characterization study complements existing work that have analytically and/or experimentally studied RTS/CTS performance. It

MobiQuitous '18, November 5–7, 2018, New York, NY, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-6093-7/18/11...$15.00

shows that network contention, as well as packet size, and transmission rate must be collectively considered in order to decide whether to enable or disable 802.11's RTS/CTS mechanism. Based on the results of our RTS/CTS performance characterization, we propose a novel approach that uses machine learning to dynamically switch RTS/CTS on and off ahead of data transmission by considering a combination of "air time", i.e. the ratio between the size of data/control information being transmitted and transmission rate, as well as network contention. It is noteworthy that (1) by accounting for network contention, the proposed mechanism is able to automatically adapt to different WLAN environments and dynamics and (2) by considering packet size and transmission rate, it will also accommodate different IEEE 802.11 variants, especially as new versions target new applications and have increasingly larger packet sizes and transmission rates. Additionally, as wireless networks become denser (e.g., in urban scenarios), the importance of avoiding potential interference amongst them grows and thus efficient use of RTS/CTS becomes even more critical to achieve adequate performance.

To the best of our knowledge, our algorithm is the first to allow automatically enabling and disabling the RTS/CTS handshake based on a set of current network conditions, and dynamically adapt when these conditions change. As current and emerging wireless network environments evolve and become increasingly more heterogeneous in terms of the underlying network technologies, connected devices, as well as driving applications, being able to dynamically adapt protocol behavior in response to changing conditions and requirements is critical to achieve adequate performance in a seamless manner. Our experimental results using both synthetic workloads as well as real traces show that our mechanism consistently outperforms current best practices, such as never enabling RTS/CTS or setting the *RTS Threshold* (*RT*), which is used to decide whether to switch RTS/CTS on or off, to a static value.

The rest of this paper is organized as follows. Section 2 provides a brief overview of IEEE 802.11 DCF and discusses RTS/CTS' tradeoffs. We discuss related work in Section 3 and in Section 4, conduct our empirical performance evaluation of 802.11's RTS/CTS as a function of different factors. Our method to dynamically enable or disable RTS/CTS is described in Section 5 and in Section 6, we evaluate the proposed approach under different network scenarios. Finally, Section 7 concludes the paper with directions for future work.

2 BACKGROUND

As previously pointed out, IEEE 802.11's Distributed Coordination Function (DCF) [1] is based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol. In DCF's *base mode*, CSMA [17] is used by stations that have data to send to check whether the shared medium is being used. More specifically, a station that wants to transmit a data frame senses the channel to check whether it is idle for a DCF Inter-frame Space (DIFS) interval. If the channel is sensed idle, the station transmits the data. Otherwise, it defers transmission using a random backoff timer. Figure 1 illustrates how DCF's base mode works. After transmitting data, the station waits for an acknowledgement (ACK). If the ACK is received, the station considers the data frame delivered. If not, it will assume a collision occurred and uses a slotted Binary Exponential Backoff (BEB) scheme to retransmit the frame at a later time.

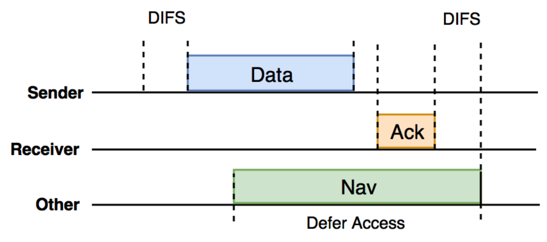

Figure 1: IEEE 802.11 Base Mode: CSMA

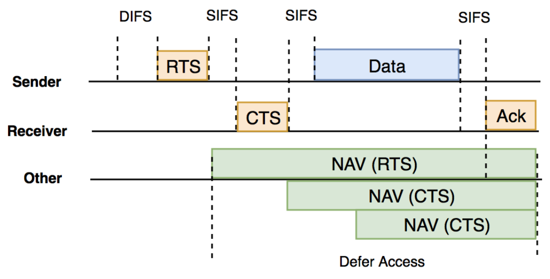

Figure 2: IEEE 802.11 CSMA/CA

IEEE 802.11's DCF can be configured to use, in addition to physical carrier sensing, virtual carrier sensing, or Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) [2]. CSMA/CA avoids hidden terminal collisions by reserving the channel ahead of data transmission. Channel reservation is achieved via a two-way handshake using small control frames, namely the Request to Send (RTS) and Clear To Send (CTS). The RTS/CTS handshake works as follows: the sender sends the RTS frame to the receiver, and the receiver responds with a CTS. Other stations that overhear either the RTS, CTS, or both mark the channel as busy and set their network allocation vector (NAV) based on the time offered in sender's RTS and/or receiver's CTS. This means that they will defer their transmissions for the interval indicated in their NAV. The sender uses the receipt of the CTS frame from the receiver as the indication that the channel has been reserved for its transmission, and therefore, the sender transmits the data frame. Figure 2 shows how the RTS/CTS handshake works.

As briefly described in Section 1, although the RTS/CTS handshake has clear benefits, it also introduces overhead. In the remainder of this section, we discuss RTS/CTS' pros and cons.

2.1 RTS/CTS Upsides

The RTS/CTS exchange was proposed as a way to combat the hidden node problem as illustrated in Figure 3. In the scenario of Figure 3, node *B* is in the transmission range of both nodes *A* and *C*. However, because *A* and *C* are outside each other's transmission range, they cannot hear each other, and thus are said to be "hidden" from one another. Suppose that DCF's base mode is used and *A* is sending a packet to *B*. Suppose that *C* also has data to send to *B*. Since *C*

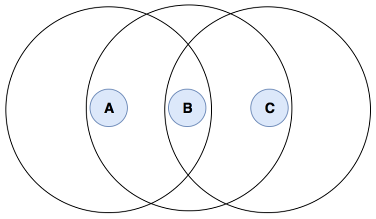

Figure 3: Hidden Node Scenario

cannot hear A, it senses the channel idle and sends data to B which results in a collision and node B discards both frames.

If RTS/CTS is used, A sends a RTS frame before the actual data. All of A's neighbors mark the channel as busy. Upon reception of the RTS, B sends a CTS back to A if it is not busy. All of B's neighbors, including C, mark the channel as busy after hearing the CTS. As a result, C defers its transmission for the time specified in the CTS frame. To show how effective RTS/CTS can be in avoiding collisions, we ran a basic experiment with four hidden nodes sending to a central node, e.g., and access point. We use four CBR streams each sending at 5.5 Mbps with 1500 byte packets. Comparing the throughput obtained with and without RTS/CTS (Table 1), we observe that RTS/CTS can yield significant performance benefits (in this scenario, more than 70%).

Table 1: Throughput Comparison: DCF's Base Mode (CSMA) versus RTS/CTS (CSMA/CA)

	Basic Mode (Mbps)	RTS/CTS Mode (Mbps)
Stream 1	0.109	0.627
Stream 2	0.124	0.595
Stream 3	0.1	0.535
Stream 4	0.134	0.568

2.2 RTS/CTS' Downsides

While the RTS/CTS mechanism can mitigate the hidden terminal problem and avoid collisions, it has some drawbacks as described below.

2.2.1 Overhead. One of the main problems is the latency and additional load that RTS and CTS control frames introduce, which, in some scenarios, outweighs RTS/CTS benefits. IEEE 802.11 defines the RTS and CTS frame sizes as 20 and 14 bytes, respectively. So, in the case of short data frames, it may not be worthwhile incurring the additional delay needed to perform the RTS/CTS exchange. That was the motivation behind proposing the *RTS Threshold*, or *RT*, which determines the minimum data frame size that will trigger the RTS/CTS handshake to reserve the channel to transmit the data frame. Frames smaller that *RT* will be sent using DCF's base mode.

2.2.2 Disabling Safe Concurrent Transmission. Another example scenario that illustrates how RTS/CTS can negatively impact throughput is as follows. Suppose that while node A is transmitting to node B, D has data to send to C and transmits an RTS frame to C. However, C is blocked because of B's CTS sent in response to

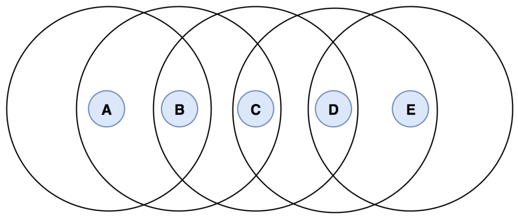

Figure 4: 5 Nodes Scenario

A's RTS to B and is not able to respond to D with a CTS. In this case, all nodes in the transmission range of D will be blocked for the time specified in D's RTS frame.

It is worth pointing out that Denial of Service attacks could be staged/caused by malicious/malfunctioning nodes who can (un)-intentionally keep sending RTS or CTS frames possibly reserving the channel for long periods of time.

3 RELATED WORK

As the de-facto standards for WLAN communication, IEEE 802.11 has been the subject of a considerable body of work both from the networking research community as well as from network practitioners. In this section, we will focus on efforts that are more closely related to our work.

In particular, a number of studies have focused on studying the performance of IEEE 802.11's DCF optional collision avoidance mechanism which uses the RTS/CTS handshake. We categorize these efforts in two groups. The first group investigates throughput performance when using or not using RTS/CTS and have advocated both for and against the use of RTS/CTS. For example, the work reported in [9], [11], [14] and [19] study the impact of using RTS/CTS when compared to DCF's base mode. In [3] and [16], RTS/CTS performance was evaluated under different data rates and it was concluded that RTS/CTS does not show much benefit at higher data rates. In a more recent study [20], it was shown that, when the network is under "stress", e.g., high node density, high traffic load, etc, the RTS/CTS threshold *RT* value can significantly impact performance in terms of end-to-end delay, medium access delay, retransmission attempts, network load, and throughput.

Other efforts have explored how to dynamically set the *RTS Threshold*, *RT*. For example, in [9], a mechanism in which the sender counts the number of "Waiting for CTS" timeouts is proposed. If this value exceeds a certain threshold, RTS/CTS is disabled. In [10], each node monitors network condition dynamics using recent packet delivery ratio measurements and adjusts the *RT* accordingly. In [13], *RT* is set based on the number of hidden terminals, while the approach described in [4], sets the *RT* based on the packet size distribution. The average number of competing terminals rather than the total number of terminals determines the *RT* value in [12]. In [15] and [16], the impact of transmission rate on RTS/CTS performance is investigated but do not use it as to disable or enable RTS/CTS. In [3], the *RT* is adjusted dynamically according to the data rate and number of stations.

As previously discussed, to our knowledge, no prior work has explored the combination of factors we consider in our approach to automatically enable or disable the RTS/CTS handshake in response to changing network- and traffic conditions. As wireless networks and protocols evolve to accommodate: (1) an ever growing number of connected devices, as well (2) as increasing heterogeneity of the underlying network technologies and (3) diversity of the driving applications, performance studies like the one we present in Section 4 below, which evaluate key protocol features such as IEEE 802.11's RTS/CTS under conditions reflecting emerging applications' and network technology trends (e.g., higher transmission rates, larger data aggregates, etc.), are critical. And, as our study indicates, introducing mechanisms such as the one we propose in Section 5 allow widely deployed protocols like IEEE 802.11 to adjust to network- and application technology trends as seamless as possible.

4 CHARACTERIZING RTS/CTS PERFORMANCE

In this section, we conduct an empirical characterization of RTS/CTS performance as a function of a number of factors, namely packet size, transmission rate for both data and signaling traffic, as well as network contention. The goal here is three-fold, namely: (1) confirm experimentally some analytical results on the performance of RTS/CTS in the literature; (2) make the case for dynamically adjusting the *RTS Threshold, RT* based on data and control transmission time as well as network contention; and (3) validate our experimental methodology and setup.

To show the effect of each factor on RTS/CTS performance, we run several simulation experiments using the *NS-3* network simulator [18] for infrastructure-based network scenarios [1]. In our experiments, we associate 5, 10, and 20 nodes to an access point (AP) which are either all hidden or not hidden from each other. We use *NS-3*'s *Matrix Propagation Loss Model* and set the propagation loss between each pair of nodes to make them hidden or not hidden from each other. For example, if we set the propagation loss between nodes *A* and *B* to a very high value, then *A* becomes hidden from *B*, and vice-versa. Nodes send CBR traffic to the AP with packet sizes of 200, 500, 1000, 1500, and 2000 bytes and data rates of 2, 5.5, 11, 24, and 54 Mbps, while signaling transmission rate is kept at 2 Mbps. The graphs shown in Figures 5, 6, and 7 plot *RTS/CTS throughput gain*, T_{Gain} which is defined as follows:

$$T_{Gain} = (T_{RTS/CTS} - T_{Base})/T_{RTS/CTS} \qquad (1)$$

where $T_{RTS/CTS}$ and T_{Base} are the throughput when using RTS/CTS and the throughput when using DCF's base mode, respectively.

4.1 Data Transmission Time

As previously pointed out, data packet transmission time, which is a function of the data packet size and the transmission rate, is an important factor affecting RTS/CTS performance. Data packet transmission time should be long enough to warrant the overhead of the RTS/CTS handshake. When data packet transmission time is comparable to the latency of the RTS/CTS exchange, there is no need to add the extra overhead of RTS/CTS since the cost of data frame collision is comparable to collision of the control frames themselves.

[1]Characterizing RTS/CTS performance for infrastructure-less network scenarios is being conducted as part of our ongoing work.

This is illustrated in Figure 5 which shows the normalized RTS/CTS throughput gain over IEEE 802.11 DCF's base mode as a function of data packet air time. As expected, for larger data packet sizes, RTS/CTS is more effective (throughput gains above 0). For smaller packets, e.g. 200 bytes, sent at high data rate, e.g. 54 Mbps, the airtime used by data packets is small enough that collision cost is negligible.

One of the current practices to decide when to use RTS/CTS employs a fixed *RTS Threshold, RT* based on packet size. In fact, it is common to set the *RT* to the maximum data packet size which results in never using RTS/CTS, which may be adequate in some scenarios, but not in others as exemplified by the results in Figure 5.

A notable practical factor in the performance of RTS/CTS, which is frequently neglected, is that, in multi-rate WLANs, control frames such as ACK, RTS, and CTS are transmitted at a fixed basic rate regardless of the data rate. One of the main reasons is to enable interoperability and to accommodate legacy devices, since all devices in the network must be able to receive these frames. This increases the overhead of the RTS/CTS mechanism in high data rate networks, which is also captured by the results in Figure 5 as the signaling rate is kept at 2 Mbps.

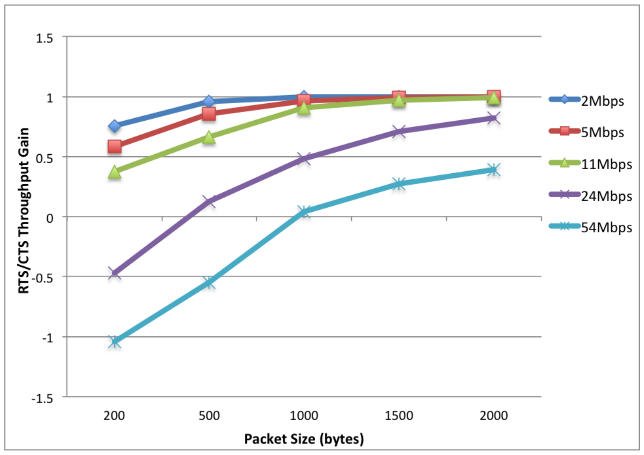

Figure 5: RTS/CTS throughput gain as a function of data packet size for different data rates.

4.2 Signaling Rate

In this section, we consider network scenarios where signaling can be transmitted at the same rate as data.

Figure 6 plots the RTS/CTS throughput gain for different signaling rates (equal to the data rate). It shows positive throughput gains across the board. These results confirm that adjusting *RT* also needs to account for transmission rate of control packets (signaling rate).

4.3 Network Contention

Network contention is clearly a critical factor in deciding whether to use RTS/CTS. Recall that the original goal of the RTS/CTS handshake is to avoid collisions, so if collisions are not likely to occur

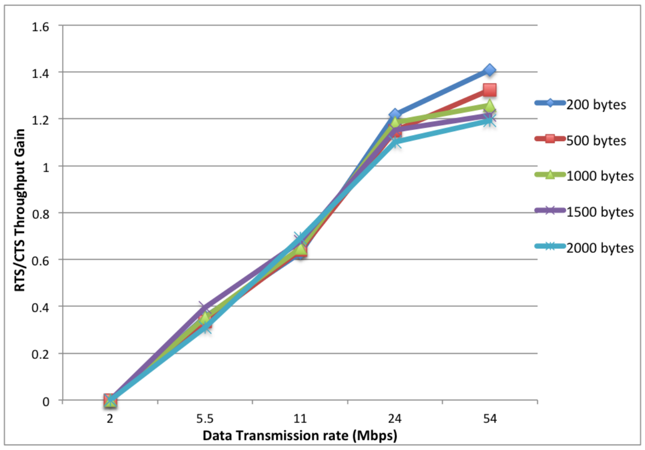

Figure 6: RTS/CTS throughput gain as a function of signaling rate for different packet sizes.

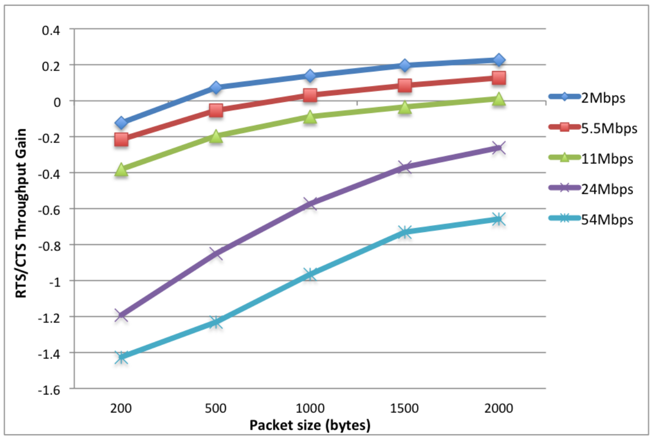

Figure 7: RTS/CTS throughput gain as a function of packet size for different data rates when no hidden nodes are present.

(e.g., low network load), there is no need to use RTS/CTS and incur the additional overhead. In IEEE 802.11 networks, collisions occur either because of the existence of hidden terminals or due to (quasi) simultaneous transmissions (e.g., as a result of back off synchronization).

As described in Section 3, there have been a number of proposals to estimate network contention, such as number of hidden terminals [13], mean medium access delay [3], packet delivery ratio [10], number of waiting for CTS [9], to name a few. We measure network contention by calculating the *collision probability*, which we describe in Section 5.1.

We conducted similar sets of experiments as the ones described in Section 4.1, except for the fact that we use a topology with no hidden nodes. As shown in Figure 7, for most data rates and small-to medium packet sizes, there is no benefit in using RTS/CTS when there is no hidden terminal. These results confirm the importance of accounting for network contention when deciding whether to use RTS/CTS.

5 PROPOSED APPROACH: DYNAMICALLY SWITCHING RTS/CTS ON-OFF

We propose a simple yet effective algorithm to dynamically enable or disable RTS/CTS. Unlike most previous efforts which typically consider individual factors in deciding when to use or not use RTS/CTS, our method considers all factors, i.e., network contention, as well as data and signaling transmission time, also known as *air time*, which is given by Equation 2 below.

$$transmission\ time = \frac{data/signaling\ size}{data/signaling\ rate} \qquad (2)$$

Algorithm 1 describes our approach which essentially evaluates on an ongoing basis the benefit of using RTS/CTS compared to its overhead. If RTS/CTS is deemed beneficial, i.e., it avoids collisions, when *collision cost* is higher than the cost of the RTS/CTS exchange, then RTS/CTS is enabled, otherwise it is disabled. Note that we define *collision cost* as a function of network contention and data transmission time, which we represent by $network\ contention \otimes data\ transmission\ time$ in Algorithm 1. *Data transmission time*, as well as RTS/CTS cost (or *signaling transmission time*) are calculated according to Equation 2. Note that data frame size, data transmission rate, signaling transmission rate, and RTS/CTS frame size (which amounts to 34 bytes) are known at the time of transmission. However, current network contention conditions must be estimated on an ongoing basis. We describe our approach for network contention estimation in Section 5.1 below.

Algorithm 1 Proposal

```
If ($network\ contention \otimes data\ transmission\ time$) $\geq$ $signaling\ transmission\ time$
then enable RTS/CTS
else disable RTS/CTS
```

5.1 Network Contention

Network contention depends on the number of competing stations that are simultaneously trying to access the shared communication medium in order to transmit. In this paper, we use *collision probability* as an indicator of network contention and measure it by dividing the number of failed receptions at the receiver by the total number of transmissions at the sending node.

As illustrated in Figure 8, we divide time into slots, and at the beginning of each slot, there is a short *learning period*, during which RTS/CTS is disabled. *Collision probability* is measured a number of times during the *learning period*. Then, using the SENSE estimator [5], described in Section 5.2 below, we estimate the collision probability for the remaining of the slot. Based on SENSE's collision probability estimate, our current implementation of Algorithm 1 calculates $network\ contention \otimes data\ transmission\ time$ as the product between *data transmission time* and collision probability. It then decides whether to turn RTS/CTS on or off for the duration of the slot. Time slot and learning period duration as well as the number of times we calculate collision probability during the learning period are parameters of our approach. In our performance evaluation (described in Section 6), we experimented with different values of these parameters and did not observe significant changes in the results. As part of our future work, we plan to explore further our mechanism's sensitivity to these parameters as well as techniques to automatically adjust them based on network behavior. We also plan to investigate alternate approaches to assess network contention (e.g., sender's MAC queue length, mean time to access medium, etc).

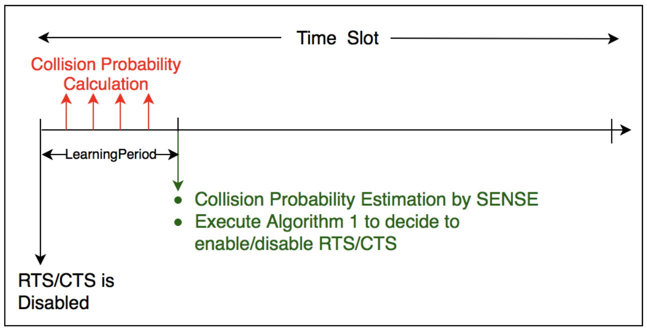
Time Slot
Collision Probability Calculation
LearningPeriod
Collision Probability Estimation by SENSE
Execute Algorithm 1 to decide to enable/disable RTS/CTS
RTS/CTS is Disabled

Figure 8: Time slot

5.2 SENSE Estimator

SENSE [5] is an online estimator, which employs a combination of Exponentially Weighted Moving Average (EWMA) and a simple machine learning algorithm known as Fixed-Share.The accuracy of this estimator is studied in [5].

EWMA based predictors, calculate an exponentially weighted mean of the data [7]. Equation 3 shows the EWMA equation where x_t is the estimated data and y_t is the data point that has been observed. α is the "smoothing factor", a value between 0 and 1 which controls how much weight is given to previous estimates (i.e., the "past") versus new samples (the "present"). The problem of EWMA-based predictors is to find the appropriate α value to use. Low values of α, favor the "past" over the "present" while with high α values, the "present" plays a more important role. Therefore, there should be a priori knowledge of data's behavior to choose an appropriate α.

$$x_t = \alpha \times y_{t-1} + (1 - \alpha) \times x_{t-1} \tag{3}$$

Fixed-share algorithms [8] are a member of the Multiplicative Weight algorithmic family. This family of algorithms combines predictions of a set of experts to compute the overall prediction. The impact of each expert on the overall predictor is determined by a weight associated with each expert. The weight of each expert is updated at the end of each trial based on the difference between its prediction and the real data. Although the Fixed-Share algorithm has been shown to perform well [6], it has some drawbacks. First, a fix value within range of data is given to each expert as its prediction. So, the range of data should be known before using these predictors. Second, Fixed-share can not adjust to abrupt changes in data fast enough because it takes a long time for the weight of an expert to either shoot up or down when the expert's performance suddenly changes following an abrupt change in the data. Third, the accuracy of algorithm is sensitive to the number of experts. More experts can cover more data from the dataset. However, it may also introduce additional error.

SENSE is a variant of the Fixed-Share algorithm, where, instead of fixed-value experts, EWMA filters are employed as experts. Algorithm 2 shows SENSE's pseudo-code and Table 2 summarizes the notation used the SENSE pseudo-code. At the initialization phase, each expert is given a weight, $w_{i,1} = 1/N$, where N is the total number of experts. Each expert is also assigned an α value between 0 and 1 which separates experts from each other. As shown in Algorithm 2, in the EWMA experts step each experts' prediction is calculated as a weighted sum of the previously seen data (y_{t-1}) and the previous prediction $x_{i,t-1}$. In the prediction step, SENSE calculates the current prediction by adding the weighted predictions from N experts. The loss function step, calculates the absolute difference between the actual data and each expert's forecast and normalizes this error with the maximum outcome y_{max}. Loss function is set to either the normalized error or the NULL function based on the normalized error. The META-learning step decides on η which helps to adjust the experts' weights based on their recent performance. The less precise an expert's prediction is, the more severe that expert is penalized. Then, SENSE updates each experts' weight with what has been learned. Finally, Level-shift step detects significant changes in the mean of the observed data and restarts its experts by only considering data after the level shift and reseting η of each expert.

```
Algorithm 2 SENSE
  Initialization:
      $w_{1,1} = ... = w_{N,1} = \frac{1}{N}$
  EWMA Experts:
      $x_{i,t} = \alpha_i \times y_{t-1} + (1 - \alpha_i) \times x_{i,t-1}$
  Prediction:
      $\hat{y}_{i,t} = \frac{\sum_1^N w_{i,t} \times x_{i,t}}{\sum_1^N w_{i,t}}$
  Loss Function:
  $NE_{i,t} = \frac{|x_{i,t} - y_t|}{y_{max}}$
      $L(x_i,t)_{i,t} = \begin{cases} NULL & , NE_{i,t} \leq EL \\ NE_{i,t} & , Otherwise \end{cases}$
  META Learning:
  $\eta_{i,t} = \begin{cases} min(\eta_m ax, (\eta_{i,t-1} \times \beta)) \\ \quad , NE_{i,t} > NE_{i,t-(j-1)} > NE_{i,t-j} \\ max(\eta_m in, (\frac{\eta_{i,t-1}}{\beta})) \\ \quad , NE_{i,t} < NE_{i,t-(j-1)} < NE_{i,t-j} \\ \eta_{i,t-1} \\ \quad , Otherwise \end{cases}$
  Weight Update:
      $w_{i,t+1} = w_{i,t} \times e^{-\eta_{i,t} \times L(x_i,t)_{i,t}}$
  Restart Learning:
      If Level Shift is detected at $nk$ then,
      $w_{i,t} = w_{i,t} \times e^{\sum_{t=-2T}^{t=-T} \eta_{i,t} \times L(x_i,t)_{i,t}}$
```

Table 2: SENSE's Variables

Parameter	Description
x_i	Prediction of expert i
y_t	Observed data at time t
$\hat{y}_t$	SENSE's prediction for time t
w_i	Weight of expert i
NE_i	Normalized error of expert i
N	Total number of experts
$L(x_i, t)_{i,t}$	Loss of expert i at time t
y_{max}	Maximum data observed so far
η_i	Determines degree of penalizing expert i
β	Determines how much η_i should be increased or decreased
EL	Error limit (based on user's desired accuracy)
η_{min}, η_{max}	Limit experts' weight
j	Determines the time window to evaluate expert's performance (used to update η_i)

6 EXPERIMENTS AND RESULTS

We ran simulations using the *NS-3* network simulator [18] and its implementation of the IEEE 802.11n. Our simulations were driven by synthetic data traces as well as traces collected in real networks. For each set of experiments, we describe the experimental methodology employed (e.g., trace generation/collection, parameter values, etc), as well as their results. For all experiments, we use time slots of 2, 5, 10, and 20 seconds, learning periods of 0.4, 1, 2, and 4 seconds, and vary the number of times the collision probability is calculated from 1 to 10 times, in increments of 1. Setting these parameters using these different values has not resulted in any significant change in our mechanism's performance. As such, for the results shown in this section, we set the time slot to 5 seconds, the learning period to 1 second, and we calculate collision probability 4 times during the learning period.

We compare the average throughput and confidence intervals over 10 runs when using our approach against the average throughput of having: RTS/CTS always disabled, and RTS/CTS always enabled. We also show the average throughput when using statically configured values for the *RTS Threshold*, RT, namely 0, 200, 500, 1000, 1500, and 2000. While we have not directly compared our approach to existing techniques that account for only a subset of the factors we consider, our results already indicate the benefits of making the decision of switching RTS/CTS on or off based on a combination of data airtime, network contention, as well as signaling airtime.

6.1 Synthetic Trace

Experimental Methodology. We use similar experimental setup as in the experiments of Section 4, where an AP is connected to 5, 10, or 20 nodes, each of which sends CBR traffic with packets of sizes 200, 500, 1000, 1500, and 2000 bytes to the AP. Half of the nodes are hidden from each other. We ran each experiment with data transmission rates of 54, 24, 11, 5.5, or 2 Mbps, while signaling transmission rate is kept at 2 Mbps.

To show how our algorithm adjusts to network dynamics, we periodically changed packet size and number of nodes transmitting to the AP during the experiments as described in the first three columns of Tables 3 and 4. Note that by varying the number of simultaneous transmitters, we vary network contention and consequently collision probability (whose values are listed in Tables 3 and 4).

Results. Figure 9 shows the average throughput resulting from using our algorithm ("Adaptive"), IEEE 802.11 DCF's base mode ("Basic"), and RTS/CTS enabled based on different RT values for the case when 20 nodes are connected to the AP. We omit the confidence intervals for Figure 9 to make the graph easier to read. We observe that our approach to dynamically switching RTS/CTS on and off outperforms all other techniques tested for all data rates used. As expected, for lower data rates, "Basic" yields the lowest performance because packet transmission takes longer and therefore collision probability is higher. In the low data rate cases, RTS/CTS enabled with lower RT values provide better performance but as the data rate increases, performance of "Basic" gets closer to RTS/CTS, and eventually outperforms it at higher data rates. For example, for 24 Mbps, RTS/CTS with mid-range RT values provide better performance than both "Basic" and RTS/CTS with larger RT values. The reason is that throughput improves when larger packets are "protected" by RTS/CTS at this data rate while, for smaller packets, throughput is higher without RTS/CTS. Our proposed algorithm performs well because it can dynamically decide when to enable or disable RTS/CTS based on current conditions, i.e., packet size, transmission rate, and contention. For instance, in the 2Mbps scenario, our approach achieves twice the throughput of "Basic" and 12% higher throughput than the second best performer in this case which is $RT = 200$. However, for 54 Mbps data rate, "Basic", which performs better than all RT values, is outperformed by our algorithm by 5%, while $RT = 200$ is outperformed by our approach by close to 40%.

Tables 3 and 4 show, on a slot-by-slot basis, the behavior resulting from our algorithm, i.e., whether RTS/CTS is enabled or disabled and under which conditions. For example, in seconds 20 to 25, where packets are 1000 bytes and collision probability is 45%, when data transmission rate is 54 Mbps, RTS/CTS is disabled. However, in 24, 11, 5.5 and 2 Mbps, RTS/CTS is used. This is because, for lower data rates, using RTS/CTS is more advantageous.

6.2 Public Hot Spot Trace

Trace Collection. To further evaluate our algorithm's ability to decide whether to use RTS/CTS or not, we drive it using real traffic traces captured in a public hot spot using a wireless sniffer. Table 5 summarizes the trace's features. Data rates provided in the *Radiotap Header* are used to calculate a packet's airtime. We captured 10 flows between users and the AP for 20 minutes and feed the flows to the *NS-3* simulator. Packet size ranges between 34 and 2150 bytes. We ran our experiment with 10, 30, and 50 nodes using *NS-3*'s IEEE 802.11n. Each flow is assigned to 1, 3, and 5 nodes in the 10, 30, and 50 nodes scenarios, respectively and has different start times. Similarly to the previous experiments, we vary the number

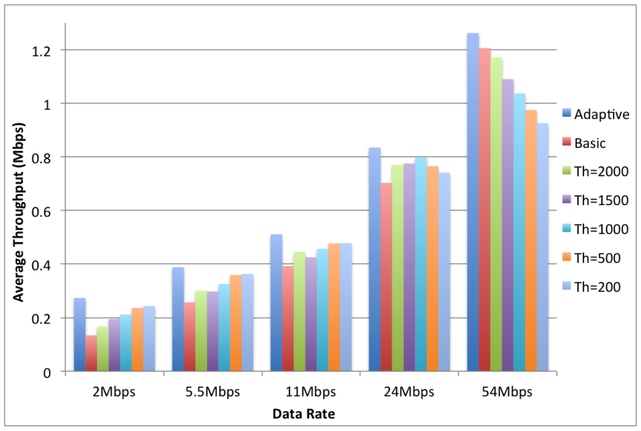

Figure 9: Average throughput for the proposed dynamic algorithm and the different static approaches to enabling/disabling RTS/CTS.

Table 3: Slot-By-Slot Behavior of Proposed Algorithm for 54 and 24 Mbps Data Rates.

Time (s)	Packet Size	Collision	54Mbps	24Mbps
0 to 5	1500	4%	Basic	Basic
5 to 10	500	12%	Basic	Basic
10 to 15	2000	25%	Basic	RTS/CTS
15 to 20	200	38%	Basic	Basic
20 to 25	1000	45%	Basic	RTS/CTS
25 to 30	2000	59%	RTS/CTS	RTS/CTS
20 to 35	500	64%	Basic	Basic
35 to 40	200	73%	Basic	Basic
40 to 45	1500	77%	RTS/CTS	RTS/CTS
45 to 50	500	79%	Basic	Basic

Table 4: Slot-By-Slot Behavior of Proposed Algorithm for 11, 5.5, and 2 Mbps Data Rates.

Time (s)	P-Size	Collision	11Mbps	5.5Mbps	2Mbps
0 to 5	1500	4%	Basic	Basic	RTS/CTS
5 to 10	500	12%	Basic	Basic	RTS/CTS
10 to 15	2000	25%	RTS/CTS	RTS/CTS	RTS/CTS
15 to 20	200	38%	Basic	Basic	RTS/CTS
20 to 25	1000	45%	RTS/CTS	RTS/CTS	RTS/CTS
25 to 30	2000	59%	RTS/CTS	RTS/CTS	RTS/CTS
20 to 35	500	64%	RTS/CTS	RTS/CTS	RTS/CTS
35 to 40	200	73%	Basic	RTS/CTS	RTS/CTS
40 to 45	1500	77%	RTS/CTS	RTS/CTS	RTS/CTS
45 to 50	500	79%	RTS/CTS	RTS/CTS	RTS/CTS

of nodes and number of concurrent flows in order to vary network contention.

Results. Similarly to the synthetic trace experiments, we compare the average throughput when using our adaptive method against

Table 5: Hot Spot Trace

Location	Coffee shop
Time	Around noon
Number of flows	10
Duration	20 minutes
Packet size	Varies within 34-2150 byte range
Number of hidden nodes	Half of the nodes
802.11 version	802.11n

the basic mode (no RTS/CTS), as well as statically configured *RT* values of 0 (RTS/CTS always enabled), 200, 500, 1000, 1500, and 2000.

Figures 10 - 12 show the average throughput for the 10, 30, and 50 node scenarios, respectively. As shown in Figure 10, in the 10-node experiment, since there is less contention, using the basic mode or higher *RT* values is more beneficial. In Figure 11, with more nodes, and as a result higher contention, lower thresholds like 200 and 500 bytes perform better. Figure 12 shows that RTS/CTS should be used all the time in the 50-node scenario because of the high contention. Our adaptive approach outperforms all other methods in all cases because of its ability to adjust to network contention and airtime. In other words, while in the 10-node experiment whose results are shown in Figure 10, "Basic" yields similar throughput when compared to our approach, in Figure 11, which shows results for the 30-node scenario, "Basic" is the worst performer and our approach is able to deliver through that is over 50% higher than "Basic"'s throughput.

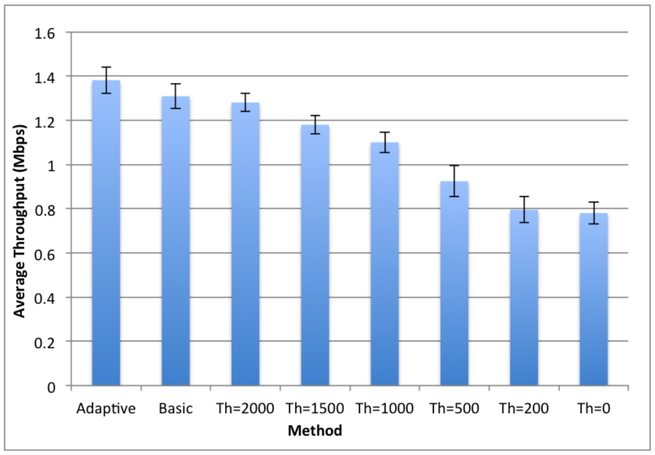

Figure 10: Average throughput using hot spot trace in 10-node scenario. 95% confidence intervals are shown.

6.3 Campus Network Trace

Trace Collection. We also evaluated our algorithm using traffic traces collected in a company's campus network. Details about the trace are listed in Table 6. We captured 5 flows between users and an AP for 30 minutes and fed them to the *NS-3* simulator. Packet

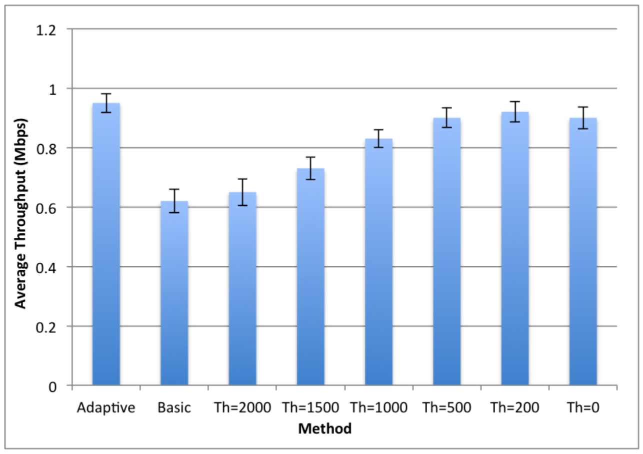

Figure 11: Average throughput using hot spot trace in 30-node scenario. 95% confidence intervals are shown.

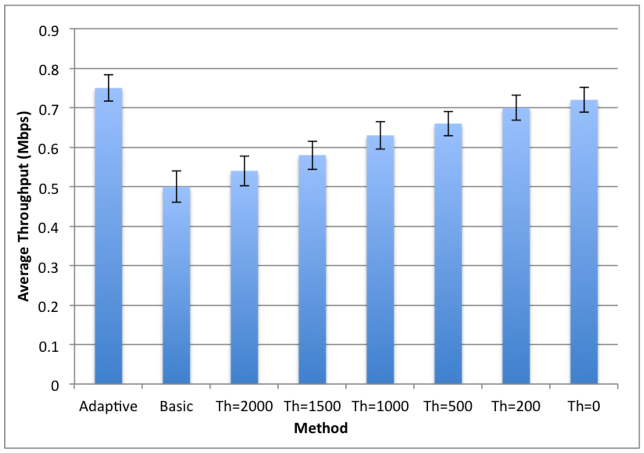

Figure 12: Average throughput using hot spot trace in 50-node scenario. 95% confidence intervals are shown.

sizes range between 34 and 11′000 bytes and half of the nodes are hidden from each other. Similarly to the hot spot trace, data rates provided in the *Radiotap Header* are used to calculate the airtime. We ran experiments with 5, 10, 30, and 50 nodes by assigning each captured flow to 1, 2, 6 and 10 nodes, respectively. Compared to the public hot spot trace, the average packet size and data rate are much higher in this dataset.

Results. This set of experiments confirms the importance of using packet size and transmission rate when deciding whether to switch RTS/CTS on or off. For instance, even though contention is not high in the 5-node scenario (Figure 13), since packets are larger on average, enabling RTS/CTS yields higher average throughput when compared to Basic. It is interesting to observe that while in Figure 13 which shows the average throughput for 5-node scenario, *RT* values of 1000 and 500 result in the highest average throughput, as the number of nodes increases (as shown in Figures 14 – 16), the optimal *RT* value decreases. So in the 50-node scenario (Figure 16) RTS/CTS

Table 6: Company Campus Network Trace

Location	Company campus network
Number of flows	5
Duration	30 minutes
Packet size	Varies within 34-11000 byte range
Number of hidden nodes	Half of the nodes
802.11 version	802.11n

should be used all the time. In all scenarios, our proposed approach outperforms all other methods since it adjusts dynamically to packet size, transmission rate and network contention.

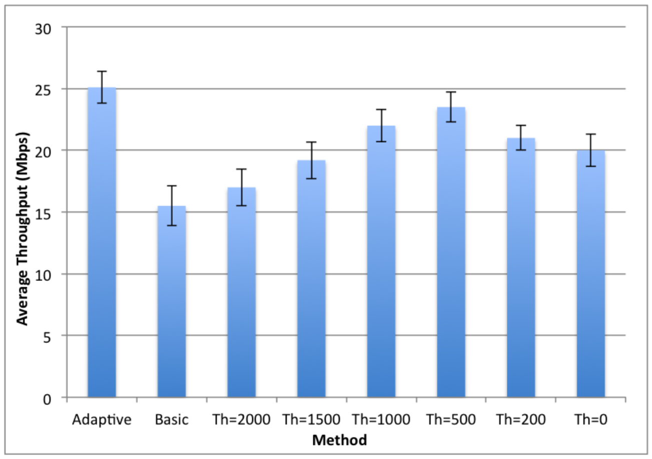

Figure 13: Average throughput using company campus trace in 5-node scenario. 95% confidence intervals are shown.

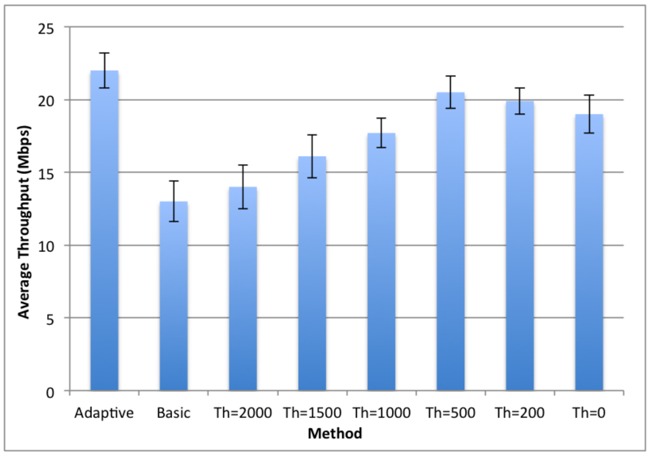

Figure 14: Average throughput using company campus trace in 10-node scenario. 95% confidence intervals are shown.

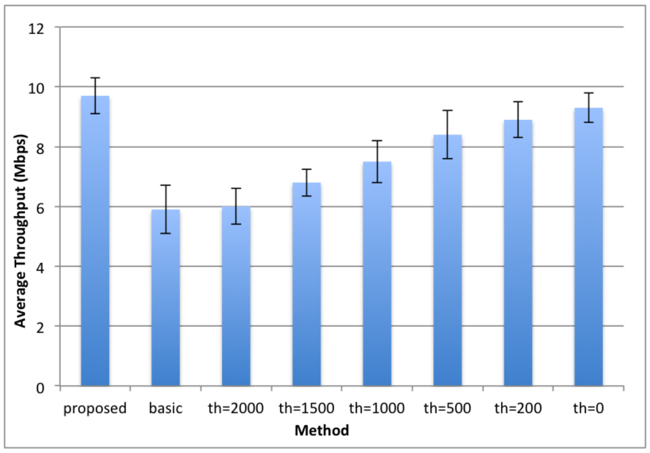

Figure 15: Average throughput using company campus trace in 30-node scenario. 95% confidence intervals are shown.

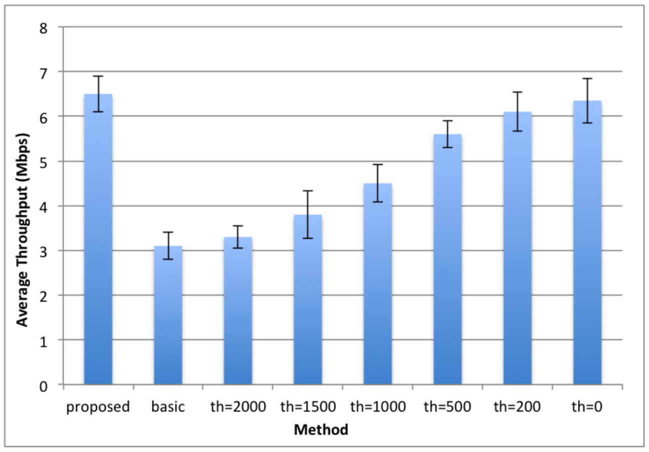

Figure 16: Average throughput using company campus trace in 50-node scenario. 95% confidence intervals are shown.

7 CONCLUSION AND FUTURE WORK

In this paper, we conducted an empirical characterization of IEEE 802.11's RTS/CTS performance as a function of packet size, transmission rate, and network contention. Based on our RTS/CTS performance characterization, we proposed a novel algorithm that dynamically decides whether to enable or disable RTS/CTS based on current network conditions and characteristics. Through simulations using a variety of WLAN network scenarios, we showed that the proposed algorithm consistently outperforms current best practice approaches which either do not enable RTS/CTS at all or use a static value of the *RTS Threshold* (*RT*) to decide whether to switch RTS/CTS on or off.

As future research directions, we plan to explore alternate techniques to estimate network contention, as well as mechanisms to set our algorithm's parameters (e.g., time slot, learning period). We also plan to adapt and evaluate the performance of our algorithm in infrastructure-less scenarios.

REFERENCES

[1] IEEE Standard 802.11 - 1999; Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications; November 1999.

[2] Phil Karn. 1990. "MACA-a new channel access method for packet radio." In ARRL/CRRL Amateur radio 9th computer networking conference, vol. 140, pp. 134–140.

[3] Zhen-ning Kong, Danny HK Tsang, and Brahim Bensaou. 2004. "Adaptive RTS/CTS mechanism for IEEE 802.11 WLANs to achieve optimal performance." In Communications, 2004 IEEE International Conference on, vol. 1, pp. 185–190. IEEE.

[4] SM Rifat Ahsan, Mohammad Saiful Islam, Naeemul Hassan, and Ashikur Rahman. 2010. "Packet distribution based tuning of RTS Threshold in IEEE 802.11." In Computers and Communications (ISCC), 2010 IEEE Symposium on, pp. 1–6. IEEE.

[5] Yalda Edalat, Jong Suk Ahn, and Katia Obraczka. 2016. "Smart Experts for Network State Estimation." IEEE Trans. Network and Service Management 13, no. 3: 622–635.

[6] Bruno Astuto Arouche Nunes, Kerry Veenstra, William Ballenthin, Stephanie Lukin, and Katia Obraczka. 2014. "A machine learning framework for TCP round-trip time estimation." EURASIP Journal on Wireless Communications and Networking 2014, no. 1: 47.

[7] J. Stuart Hunter. 1986. "The exponentially weighted moving average." Journal of quality technology 18, no. 4: 203–210.

[8] Mark Herbster and Manfred K. Warmuth. 1998. "Tracking the best expert." Machine learning 32, no. 2: 151–178.

[9] Laura Huei-jiun Ju and Izhak Rubin. 2003. "The Effect of Disengaging RTS/CTS Dialogue in IEEE 802.11 MAC Protocol." In International Conference on Wireless Networks, pp. 632–638.

[10] Mostafa Mjidi, Debasish Chakraborty, Naoki Nakamura, Kazuhide Koide, Atushi Takeda, and Norio Shiratori. 2008. "A new dynamic scheme for efficient RTS threshold handling in wireless networks." In Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on, pp. 734–740. IEEE.

[11] Kaixin Xu, Mario Gerla, and Sang Bae. 2002 "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks." In Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE, vol. 1, pp. 72–76. IEEE.

[12] Giuseppe Bianchi and Ilenia Tinnirello. 2003. "Kalman filter estimation of the number of competing terminals in an IEEE 802.11 network." In INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, vol. 2, pp. 844–852. IEEE.

[13] Tetsuya Shigeyasu, Makoto Akimoto, and Hiroshi Matsuno. 2011. "Throughput improvement of ieee802.11 dcf with adaptive rts/cts control on the basis of existence of hidden terminals." In 2011 International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 46–52. IEEE.

[14] Ashikur Rahman and Pawel Gburzynski. 2006. "Hidden problems with the hidden node problem." In Communications, 2006 23rd Biennial Symposium on, pp. 270–273.

[15] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas. 2004. "Optimisation of RTS/CTS handshake in IEEE 802.11 Wireless LANs for maximum performance." In Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004. IEEE, pp. 270–275.

[16] Ilenia Tinnirello, Sunghyun Choi, and Youngsoo Kim. 2005. "Revisit of RTS/CTS exchange in high-speed IEEE 802.11 networks." In World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a, pp. 240–248.

[17] Leonard Kleinrock, and Fouad Tobagi. 1975. "Packet switching in radio channels: Part I–Carrier sense multiple-access modes and their throughput-delay characteristics." IEEE transactions on Communications 23, no. 12 (1975): 1400–1416.

[18] The Network Simulator: NS-3: notes and documentation: https://www.nsnam.org.

[19] M. A. Khan, Tazeem Ahmad Khan, and M. T. Beg. 2012. "RTS/CTS mechanism of MAC layer IEEE 802.11 WLAN in presence of hidden nodes." International Journal of Engineering and Innovative Technology (IJEIT) Volume 2.

[20] Harsukhpreet Singh, Amandeep Kaur, Anurag Sharma, and Vishal Sharma. 2015. "Performance Optimization of DCF-MAC Standard using Enhanced RTS Threshold under impact of IEEE 802.11 n WLAN." In Advanced Computing and Communication Technologies (ACCT), 2015 Fifth International Conference on, pp. 421–424.