Threshold Optimization for Rate Adaptation Algorithms in IEEE 802.11 WLANs

Yang Song, Xiaoyan Zhu, Yuguang Fang, and Hailin Zhang

***Abstract*—Rate adaptation algorithms play a crucial role in IEEE 802.11 WLANs. While the network performance depends greatly on the rate adaptation algorithms, the detailed implementation is left to vendors. Due to its simplicity and practicality, threshold-based rate adaptation algorithms are widely adopted in commercial IEEE 802.11 devices. Taking the popular ARF algorithm for example, the data rate is increased when ten consecutive transmissions are successful and a date rate downshift is triggered by two consecutive failed transmissions. Although widely deployed, the optimal selection of the up/down thresholds for the rate adaptation algorithms remains an open problem. In this paper, we first investigate the threshold-based rate adaptation algorithm via a reverse engineering approach where the implicit objective function is revealed. Next, we propose a threshold optimization algorithm which can dynamically adjust the up/down thresholds and converge to the stochastic optimum solution in arbitrary stationary random channel environment. The performance enhancement by tuning the thresholds optimally is validated by simulations.**

***Index Terms*—IEEE 802.11 WLANs, rate adaptation, reverse engineering, learning algorithms.**

I. Introduction

IEEE 802.11 WLAN has become the dominating technology for indoor wireless Internet access. While the original IEEE 802.11 standard only provides two physical data rates (1 Mbps and 2 Mbps), the current IEEE standard provides several available data rates based on different modulation and coding schemes. For example, IEEE 802.11b supports 1 Mbps, 2 Mbps, 5.5 Mbps and 11 Mbps and IEEE 802.11g provides 12 physical data rates up to 54 Mbps. In order to maximize the network throughput, IEEE 802.11 devices, i.e., stations, need to adaptively change the data rate to combat with the time-varying channel environments. For instance, when the channel is good, a high data rate which usually requires higher SNR (signal-to-noise ratio) can be utilized. On the contrary, a low data rate which is error-resilient might be favorable for a bad channel. The operation of dynamically selecting data rates in IEEE 802.11 WLANs is called *rate adaptation* in general.

Manuscript received March 30, 2009; revised August 27, 2009; accepted October 24, 2009. The associate editor coordinating the review of this paper and approving it for publication was J. Lu.

Y. Song and Y. Fang are with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611. Y. Fang is also a Changjiang Scholar Chair Professor with the National Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China (e-mail: yangsong@ufl.edu, fang@ece.ufl.edu).

X. Zhu and H. Zhang are with the National Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China. X. Zhu is currently a Visiting Research Scholar with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 (e-mail: xyzhu@mail.xidian.edu.cn, hlzhang@xidian.edu.cn).

This work was supported in part by the U.S. National Science Foundation under Grants CNS-0916391, CNS-0721744 and CNS-0626881. The works of X. Zhu, Y. Fang and H. Zhang were also partially supported by the 111 project under grant B08038 with Xidian University, Xi'an, China.

Digital Object Identifier 10.1109/TWC.2010.01.090459

The implementation of rate adaptation algorithms is not specified by the IEEE 802.11 standard. This intentional omission flourishes the studies on this active area where a variety of rate adaptation algorithms have been proposed [1]–[17]. The common challenge of rate adaptation algorithms is how to match the unknown channel condition optimally such that the network throughput is maximized. According to the methods of estimating channel conditions, rate adaptation algorithms can be divided into two major categories. The first one is called *closed-loop* rate adaptation. Most schemes in this approach enable the receiver to measure the channel quality and sends back this information explicitly to the transmitter for rate adaptation. For example, the receiver records the SNR or RSSI (received signal strength indication) value of the received packet and sends this information back to the transmitter via CTS or ACK packet. Consequently, the transmitter estimates the channel condition based on the feedback signal and adjusts the data rate accordingly. By utilizing additional feedback mechanisms, the close-loop rate adaptation algorithms can achieve a better performance than the open-loop counterpart. However, in practice, the close-loop rate adaptation algorithms are rarely used in commercial IEEE 802.11 devices. This is because that the extra feedback information needs to be conveyed reliably and hence an inevitable modification on the current IEEE 802.11 standard is needed. This non-compatibility hinders the close-loop rate adaptation algorithms from practical implementations in current off-the-shelf IEEE 802.11 products.

The second category of rate adaptation algorithms, which is predominantly adopted by the vendors, is labeled as *open-loop* algorithms. The widely utilized Auto Rate Fallback algorithm, a.k.a., ARF, falls into this category. As many other open-loop rate adaptation algorithms, ARF adjusts the date rate solely based on the IEEE 802.11 ACK packets. For example, in Enterasys RoamAbout IEEE 802.11 card [18], two consecutive frame transmission failures, indicated by not receiving ACKs promptly, induces a rate downshift, while ten consecutive successful frame transmissions triggers a rate upshift [19]. Most commercialized IEEE products follow this up/down scheme [4], [5]. In this paper, we focus on the open-loop rate adaptation algorithms due to the practical merits. More specifically, we consider a threshold-based rate adaptation algorithm which works as follows. If there are θ_u consecutive successful transmissions, the data rate is upgraded to the next level. On the other hand, if θ_d consecutive transmissions failed, a rate downshift is triggered. Since ARF is merely a special case of this threshold-based rate adaptation algorithm, our analysis can be applied to ARF and its variants as well.

As a tradeoff with simplicity, there are several challenges existing for the threshold-based rate adaptation algorithm. First, due to the trial-and-error based up/down mechanism, it inherently lacks the capability of capturing short dynamics of the channel variations [5]. To tackle this issue, Qiao *et.al.* propose a fast responsive rate adaptation solution in [1]. By introducing a measure of "delay factor", the responsiveness of the threshold-based rate adaptation algorithm can be guaranteed. The second challenge of the threshold-based rate adaptation algorithm is the indifference to collision-induced failures and noise-induced failures. It is worth noting that in a multiuser WLAN, an ACK timeout can be ascribed to either an MAC layer collision, or an erroneous channel. However, the threshold-based rate adaptation algorithm is unable to distinguish them effectively. Therefore, excessive collisions may introduce unnecessary rate degradations which significantly deteriorate the system performance. Attributing the actual reason of a transmission failure, or a packet loss, is named *loss diagnosis* [20] and has attracted tremendous attention from the community. For example, Choi *et.al.* [21] propose an algorithmic solution, which is specific to the threshold-based rate adaptation algorithm, to mitigate the collision effect in multiuser IEEE 802.11 WLANs. Therefore, the performance deterioration by the "indifference to collisions" can be compensated effectively.

While the first two challenges of the threshold-based rate adaptation algorithm have been tackled effectively, the third major obstacle, namely, the optimal selection of the up/down thresholds, remains as an open problem. A systematic treatment on how to select the values of θ_u and θ_d in the threshold-based rate adaptation algorithm is lacking in the literature, although several heuristic solutions are proposed [2], [3].

The contribution of this paper is twofold. First, we analytically investigate the behavior of the threshold-based rate adaptation algorithm from a reverse engineering perspective. In other words, we answer the essential yet unresolved question, i.e., *"What is the threshold-based rate adaptation algorithm actually optimizing?"*, by unveiling the implicit objective function. As a result, several intuitive observations of the threshold-based rate adaptation algorithm can be explained straightforwardly by inspecting this objective function. Therefore, our reverse engineering model provides an alternative means to understand the threshold-based rate adaptation algorithm. Our work is a complement to the recent trend of reverse engineering studies on existing heuristics-based networking protocols, such as TCP (transport layer) [22]–[24], BGP (network layer) [25] and random access MAC protocol (data link layer) [26]. To the best of our knowledge, this is the first work of studying threshold-based rate adaptation algorithms from a reverse engineering perspective. Our results explicitly show that the values of θ_u and θ_d play an important role in the objective function and thus the network performance hinges largely on the selection of the up/down thresholds. In light of this, we propose a threshold optimization algorithm which dynamically tunes the up/down thresholds of the threshold-based rate adaptation algorithm and provably converges to the stochastic optimum solution in arbitrary stationary random environments. We show that the optimal selection of the thresholds significantly enhances the system's performance.

The rest of paper is organized as follows. Section II briefly overviews the state-of-the-art rate adaptation algorithms in the literature. The reverse engineering model of the threshold-based rate adaptation algorithm is derived in Section III. In Section IV, the threshold optimization algorithm is proposed. The performance evaluations are provided in Section V and Section VI concludes this paper.

II. Related Work

RBAR [12] proposes an SNR-based close-loop rate adaptation algorithm where the rate decision relies on the feedback signal from the receiver. Specifically, the receiver estimates the channel condition and determines a proper rate via the RTS/CTS exchange. While consistently outperforms the open-loop rate adaptation algorithms such as ARF, RBAR is incompatible with the current IEEE 802.11 standard by altering the CTS frames [5]. In [14], a hybrid rate adaptation algorithm with SNR-based measurements is proposed, where the measured SNR is utilized to bound the range of feasible settings and thus shortens the response time to channel variations. [6] is another example of the close-loop rate adaptation algorithms which attempts to improve the throughput by predicting the channel coherence time, which is nevertheless difficult in practice. Chen *et.al.* introduce a probabilistic-based rate adaptation for IEEE 802.11 WLANs. In [9], a rate-adaptive acknowledgement based rate adaptation algorithm is introduced. The basic idea is that by varying the ACK transmission rate, the appropriate rate information is conveyed to the transmitter. Wang and Helmy [17] propose a traffic-aware rate adaptation algorithm which explicitly relates the background traffic to the rate selection problem. However, aforementioned solutions either rely on altering the frame structures, or do not conform to the *de facto* IEEE 802.11 standard in commercial usage[1]. CHARM [10] avoids the overhead of RTS/CTS by leveraging the channel reciprocity. However, modifications on the IEEE 802.11 standardized frame structures, e.g., beacons and probe signals, are still needed.

In light of the complexity and the incompatibility of the close-loop rate adaptation algorithms, alternative open-loop rate adaptation algorithms are proposed. Although usually providing inferior performance than the close-loop solutions, open-loop algorithms soon become the predominant technique in commercial IEEE 802.11 devices due to the simplicity and the compatibility. The most popular open-loop rate adaptation algorithm is the ARF protocol proposed by Ad and Leo [13] where a rate upshift is triggered by ten consecutive successful transmission while two consecutive failures induce a rate downshift. SampleRate [11] is another widely adopted open-loop rate adaptation algorithm which performs arguably the best in static settings [5]. However, as pointed out by [5], SampleRate suffers from significant packet losses in fast changing channels. ONOE [15] is a credit-based mechanism included in MadWiFi drivers. The credit is determined by the

[1]For example, they largely rely on the RTS/CTS signaling mechanism which is hardly used in practice.

number of successful transmissions, erroneous transmissions and retransmissions jointly. However, it is pointed in [7] that ONOE is less sensitive to individual packet failure and behaves over-conservatively.

As mentioned previously, one of the drawbacks of the simple open-loop rate adaptation algorithms is the inability to discriminate the collision-induced losses and the noise-induced losses. While several collision-aware rate adaptations are available in the literature [4], [5], [8], [16], they base on the close-loop solutions which utilize either RTS/CTS signaling or modifications on the standard. In [19], [21], Choi *et.al.* propose a collision mitigation algorithm for open-loop ARF algorithm based on a Markovian modeling. However, the channel conditions are assumed to be constant in their work. In this paper, on the contrary, we particularly focus on the threshold optimization to combat with fast channel fluctuations rather than collisions. Therefore, in tandem with [19], [21], our work provide a solution to jointly mitigate the channel fluctuations and the multiuser collisions. For example, the up/down thresholds, i.e., θ_u and θ_d can be first calculated by our threshold optimization algorithm in Section IV. Next, they are subjected to further adjustments following [21] to mitigate the collision effects.

III. Reverse Engineering for the Threshold-Based Rate Adaptation Algorithm

We consider a station in a multi-rate IEEE 802.11 WLAN. There are N stations in the WLAN where each station, say i, has a transmission probability of p_i. Note that the equivalence of the p-persistent model and the IEEE 802.11 binary exponential backoff CSMA/CA model has been extensively studied in [26] and [27]. Throughout this paper, we assume that the transmission probability of each station is fixed. The interaction of the data rate and the transmission probability remains as future research. Without loss of generality, we assume that the stations have a same transmission probability of p.

In this work, we focus on the widely deployed open loop threshold-based rate adaptation algorithm. Not surprisingly, the speed of channel variations has a great impact on the performance of the rate adaptation algorithm. While the original intention of the threshold-based rate adaptation algorithm is to maximize the average throughput, a natural question arises that whether this is indeed the case. In this section, to better understand the impact of θ_u and θ_d on the performance of the threshold-based rate adaptation algorithm, we investigate the threshold-based rate adaptation algorithm via a reverse engineering approach. We assume that the RTS/CTS signals are turned off. In a time slot, say t, we denote the channel state[2] as $s(t)$ and denote the successful transmission probability, given the current transmission rate $r(t)$ and the channel condition $s(t)$, as

$$P_S\left(s(t), r(t)\right) = p(1-p)^{N-1}\left(1 - e(s(t), r(t))\right) \quad (1)$$

where e denotes the frame error rate (FER) and is given by

$$e(s(t), r(t)) = 1 - \left(1 - P_e(s(t), r(t))\right)^L \quad (2)$$

[2]Note that the number of feasible channel states can be potentially infinite.

and $P_e(s(t), r(t))$ is the bit error rate (BER) which is determined by the current data rate, i.e., modulation scheme, and the current channel condition. L is the frame length of the packet. Similarly, we define $P_F(s(t), r(t)) = 1 - P_S(s(t), r(t))$ as the transmission failure probability at time t. It is worth noting that both P_S and P_F are functions of the current data rate $r(t)$ as well as the instantaneous channel condition $s(t)$, which is random. Particularly, we assume that the threshold-based rate adaptation algorithm will increase the data rate by an amount of δ if there are θ_u consecutive successful transmissions and decrease it by δ if there are θ_d consecutive failures. Denote $u = \theta_u - 1$ and $d = \theta_d - 1$ for notation succinctness. We define a binary indicator function $\zeta(t)$ where $\zeta(t) = 1$ means that the transmission at time slot t is successful and $\zeta(t) = 0$ otherwise. Mathematically, the updating rule of the threshold-based rate adaptation algorithm can be written as

$$\begin{aligned} r(t+1) \;=\;& (r(t)+\delta)\Gamma_{\zeta(t)=1}\Gamma_{\zeta(t-1)=1}\cdots\Gamma_{\zeta(t-u)=1} \\ &+(r(t)-\delta)\Gamma_{\zeta(t)=0}\Gamma_{\zeta(t-1)=0}\cdots\Gamma_{\zeta(t-d)=0} \\ &+r(t)\Gamma_{o.w.} \end{aligned} \quad (3)$$

where

$$\Gamma_x = \begin{cases} 1, & \text{if event x is true} \\ 0, & \text{if event x is false.} \end{cases} \quad (4)$$

For example, $\Gamma_{\zeta(t)=1} = 1$ if the transmission at time t is successful and $\Gamma_{\zeta(t)=1} = 0$ otherwise. The symbol of $o.w.$ denotes the event that neither θ_u consecutive successful nor θ_d consecutive failed transmissions happened. For simplicity, we assume that the maximum allowable data rate is sufficiently large and the minimum data rate is zero, i.e., not transmitting. Define

$$\mathbf{h}(t) = [r(t), r(t-1), \cdots, r(1), e(s(t), r(t)), \cdots, e(s(1), r(1))] \quad (5)$$

as the *history vector*. In addition, we define

$$Z(t+1) = \mathbb{E}\left\{r(t+1)|\mathbf{h}(t)\right\} \quad (6)$$

where $\mathbb{E}$ is the expectation operator.

Condition 1: (C.1) The channel states between two consecutive successful transmissions or two consecutive failed transmissions are independent random variables.

We emphasize that the restrictive condition (C.1) is not our general assumption in the paper. If (C.1) is satisfied, however, the derivation of the reverse engineering analysis can be presented in a more concise form, as will be shown shortly.

First, we obtain

$$\begin{aligned} &\mathbb{E}\left\{\Gamma_{\zeta(t)=1}\Gamma_{\zeta(t-1)=1}\cdots\Gamma_{\zeta(t-u)=1}|\mathbf{h}(t)\right\} \\ &= \Pr\left\{\Gamma_{\zeta(t)=1}=1, \Gamma_{\zeta(t-1)=1}=1, \cdots, \Gamma_{\zeta(t-u)=1}=1|\mathbf{h}(t)\right\} \end{aligned} \quad (7)$$

If condition (C.1) is satisfied, (7) can be further decomposed as

$$\begin{aligned}
&\mathbb{E}\left\{\Gamma_{\zeta(t)=1}\Gamma_{\zeta(t-1)=1}\cdots\Gamma_{\zeta(t-u)=1}|\mathbf{h}(t)\right\} \\
=\;& \Pr\left\{\Gamma_{\zeta(t)=1}=1|\mathbf{h}(t)\right\}\times\Pr\left\{\Gamma_{\zeta(t-1)=1}=1|\mathbf{h}(t)\right\} \\
&\cdots\times\Pr\left\{\Gamma_{\zeta(t-u)=1}=1|\mathbf{h}(t)\right\} \\
=\;& \prod_{k=0}^{u} P_S(s(t-k),r(t-k)) \qquad (8)
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
&\mathbb{E}\left\{\Gamma_{\zeta(t)=0}\Gamma_{\zeta(t-1)=0}\cdots\Gamma_{\zeta(t-d)=0}|\mathbf{h}(t)\right\} \\
&=\Pr\left\{\Gamma_{\zeta(t)=0}=1,\Gamma_{\zeta(t-1)=0}=1,\cdots,\Gamma_{\zeta(t-u)=0}=1|\mathbf{h}(t)\right\} \qquad (9)
\end{aligned}$$

If (C.1) is assumed to be valid, we can obtain

$$\begin{aligned}
&\mathbb{E}\left\{\Gamma_{\zeta(t)=0}\Gamma_{\zeta(t-1)=0}\cdots\Gamma_{\zeta(t-d)=0}|\mathbf{h}(t)\right\} \\
&=\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k)) \qquad (10)
\end{aligned}$$

For notation succinctness, we will temporarily assume that (C.1) is satisfied. The condition will be relaxed after the implicit objective function is revealed. Therefore, we can write (6) as

$$\begin{aligned}
&Z(t+1)=\mathbb{E}\left\{r(t+1)|\mathbf{h}(t)\right\} \\
=\;& (r(t)+\delta)\prod_{k=0}^{u}P_S(s(t-k),r(t-k)) \\
&+(r(t)-\delta)\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k)) \\
&+r(t)\left(1-\prod_{k=0}^{u}P_S(s(t-k),r(t-k))\right. \\
&\left.-\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k))\right) \\
=\;& r(t)+\delta\left(\prod_{k=0}^{u}P_S(s(t-k),r(t-k))\right. \\
&\left.-\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k))\right) \qquad (11)
\end{aligned}$$

Let us revisit (3), which can be rewritten as

$$\begin{aligned}
r(t+1)=r(t)+\delta&\left(\frac{1}{\delta}\times\Big((r(t)+\delta)\Gamma_{\zeta(t)=1}\cdots\Gamma_{\zeta(t-u)=1}\right. \\
&+(r(t)-\delta)\Gamma_{\zeta(t)=0}\Gamma_{\zeta(t-1)=0}\cdots\Gamma_{\zeta(t-d)=0} \\
&\left.+r(t)\Gamma_{o.w.}-r(t)\Big)\right) \\
&=r(t)+\delta\xi(t) \qquad (12)
\end{aligned}$$

where

$$\begin{aligned}
\xi(t)\;=\;&\frac{1}{\delta}\times\Big\{(r(t)+\delta)\Gamma_{\zeta(t)=1}\Gamma_{\zeta(t-1)=1}\cdots\Gamma_{\zeta(t-u)=1} \\
+\;&(r(t)-\delta)\Gamma_{\zeta(t)=0}\Gamma_{\zeta(t-1)=0}\cdots\Gamma_{\zeta(t-d)=0} \\
+\;&r(t)\Gamma_{o.w.}-r(t)\Big\}. \qquad (13)
\end{aligned}$$

It should be noted that

$$\begin{aligned}
\mathbb{E}\{\xi(t)|\mathbf{h}(t)\}\;=\;&\prod_{k=0}^{u}P_S(s(t-k),r(t-k)) \\
-\;&\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k)). \qquad (14)
\end{aligned}$$

Therefore, we observe that (12) is a form of stochastic approximation with respect to (11). Next, we present the reverse engineering theorem for the threshold-based rate adaptation algorithm.

Theorem 1: The threshold-based rate adaptation algorithm of (3) is a stochastic approximation which solves an implicit objective function $U(t)$, with a constant stepsize of δ, where $U(t)$ is in the form of

$$\begin{aligned}
U(t)=r(t)&\left\{\prod_{k=0}^{u}P_S(s(t-k),r(t-k))\right. \\
&\left.-\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k))\right\}+\mathcal{K} \qquad (15)
\end{aligned}$$

if condition (C.1) is satisfied and $\mathcal{K}$ is a constant with respect to rate $r(t)$.

Proof: Theorem 1 follows directly from the previous analysis. Note that the threshold-based rate adaptation algorithm can be written as

$$r(t+1)=r(t)+\delta\xi(t) \qquad (16)$$

where $\xi(t)$ is the stochastic gradient and satisfies

$$\mathbb{E}\{\xi(t)|\mathbf{h}(t)\}=\frac{\partial U}{\partial r(t)}\Big|_{\mathbf{h}(t)} \qquad (17)$$

Hence Theorem 1 holds. ■

<u>REMARK</u> *1:* If condition (C.1) does not hold, we can replace

$$\prod_{k=0}^{u}P_S(s(t-k),r(t-k))$$

and

$$\prod_{k=0}^{d}(1-P_S(s(t-k),r(t-k))$$

in (15) with

$$\Pr\left\{\Gamma_{\zeta(t)=1},\cdots,\Gamma_{\zeta(t-u)=1}|\mathbf{h}(t)\right\} \qquad (18)$$

and

$$\Pr\left\{\Gamma_{\zeta(t)=0},\cdots,\Gamma_{\zeta(t-d)=0}|\mathbf{h}(t)\right\} \qquad (19)$$

respectively and the theorem remains valid.

<u>REMARK</u> *2:* It is worth noting that the objective function $U(t)$ is a time-varying function which is determined by the data rate as well as the channel conditions of the past τ time slots where $\tau=\max(\theta_u,\theta_d)$. Note that the data rate within the last τ time slots always remains unchanged. However, the channel fluctuations affect the successful transmission probability P_S and thus alter the objective function $U(t)$.

<u>REMARK</u> *3:* The partial derivative of the objective of $U(t)$

given $\mathbf{h}(t)$, i.e.,

$$\widetilde{\xi(t)} = \prod_{k=0}^{u} P_S(s(t-k), r(t-k)) - \prod_{k=0}^{d} (1 - P_S(s(t-k), r(t-k)) \quad (20)$$

is also time-varying. If the probability that the last θ_u transmissions are all successful is greater than the probability that the last θ_d transmissions are all failures, the station tends to increase the data rate and vice versa. The speed of rate increasing or deceasing is determined by the difference of these two probabilities. In other words, the partial derivative in (20) could be either positive or negative which corresponds to a rate upshift or a rate downshift. The absolute value of the instantaneous derivative determines the speed of rate changing.

REMARK *4:* A direct computation of the partial derivative in (20) is challenging, if not impossible, due to the uncertainty induced by the unpredictable stochastic channel. Therefore, the threshold-based rate adaptation algorithm, described in (3), utilizes an alternative stochastic approximation approach with the unbiased estimation of $\widetilde{\xi(t)}$, i.e., $\xi(t)$ in (13), which significantly reduces the computational complexity since only local information based on ACKs is required. This nature of simplicity and practicability facilitates the popularity of the threshold-based rate adaptation algorithm and its variants such as ARF.

To achieve a better understanding of (15), let us consider the following extreme cases from a reverse engineering standpoint.

- $(u = 0, d = 0) \Rightarrow (\theta_u = \theta_d = 1)$: In this case, the threshold-based rate adaptation algorithm increases the data rate if the current transmission is successful and decreases otherwise. From (15), we can observe that this aggressive algorithm merely compares the successful probability with the failure probability of the current time slot and expects that the next time slot will remain the current channel condition.
- $(u = +\infty, d = 0) \Rightarrow (\theta_u = +\infty, \theta_d = 1)$: From (15), we can see that the derivative is always negative since the first part of (20) is zero. Therefore, the threshold-based rate adaptation will keep decreasing data rate until the minimum data rate is reached, i.e., zero, which is consistent with the intuition.
- $(u = 0, d = +\infty) \Rightarrow (\theta_u = 1, \theta_d = +\infty)$: In this scenario, the second part of (20) is always zero and thus the algorithm will keep upgrading data rate until the maximum data rate is achieved.
- $(u = +\infty, d = +\infty) \Rightarrow (\theta_u = \infty, \theta_d = +\infty)$: The derivative of (20) is always zero and hence the data rate never changes.

Hence, by inspecting (15) and (20) directly, we provide an alternative means to understand the behavior of the threshold-based rate adaptation algorithm. In (3), we have assumed a constant stepsize δ while in current off-the-shelf IEEE 802.11 devices, a discrete set of data rates are provided. However, continuous rate adaptation can be achieved by controlling the transmission power jointly or deploying Adaptive-Coding-and-Modulation (ACM) capable devices. Therefore, our reverse engineering model, while fits in continuous rate scenarios, provides an approximate model for discrete rate adaptation scenarios. We believe that the reverse engineering result in this paper provides a first step towards a comprehensive understanding on the good-yet-simple rate adaptation algorithm designs. In addition, based on the unveiled implicit objective function of (15), the interactions of rate adaptations among multiple IEEE 802.11 stations can be investigated in a game theoretical framework.

It is immediate to observe from (15) and (20) that the selection of θ_u and θ_d has significant impact on the performance of the rate adaptation algorithm. Ideally, the rate adaptation algorithm attempts to find the data rate which maximizes the expected throughput in the next time slot, i.e.,

$$r' = \text{argmax}_r r \times P_S(s(t+1), r). \quad (21)$$

Define

$$V = r \times P_S(s(t+1), r). \quad (22)$$

Therefore, if we can estimate the value of $\frac{\partial V}{\partial r}$ and relate it by

$$\widetilde{\xi(t)} \rightarrow \frac{\partial V}{\partial r}, \quad (23)$$

then the threshold-based rate adaptation algorithm is indeed optimizing the expected throughput via the stochastic approximation approach. However, to achieve a derivative estimation of a general non-Markovian system is non-trivial [28] [29]. Moreover, if the channel appears totally random, e.g., non-stationary and fast fading, there exists no effective optimization-based solution unless certain level of knowledge on the channel randomness is available. Therefore, in the next section, we will consider a stationary random channel environment. Nevertheless, the stochastic channel can be slow-varying or fast-varying following arbitrary probabilistic distributions. We propose a threshold optimization algorithm which provably converges to the stochastic optimum values of θ_u and θ_d and the overall performance of the network is remarkably enhanced.

IV. THRESHOLD OPTIMIZATION ALGORITHM

In this section, we model the stochastic channel as a stationary random process denoted by $s(t)$. It is worth noting that if the channel is quasi-static, i.e., $s(t)$ is a piecewise constant function, the probing-based rate adaptation algorithms, e.g., SampleRate [11], can achieve good performance. It is interesting to observe that even in a quasi-static environment, for different channel conditions, the optimal values of θ_u and θ_d may be different. One feasible way to find the optimum values of θ_u and θ_d with different channel conditions, in a quasi-static environment, is the sample-path based policy iteration approach introduced in [30] and a recent survey [31], based on the Markovian model of the threshold-based rate adaptation algorithm proposed in [19].

However, it is arguable that the channel environment is stochastic and time-varying in nature. Therefore, it is not unusual that the channel condition has already changed before the optimization algorithm has reached a steady state solution. The algorithm will be thereby consistently chasing after a moving object and thus the thresholds are always chosen suboptimally. This is more severe in a fast fading stochastic environment. In light of this, we alternatively pursue *the stochastic optimum values* of the thresholds in a time-varying and potentially fast changing channel environment. That is to say, we attempt to find the set of values for θ_u and θ_d which maximize the *expected* system performance with respect to the random channel. Before elaborating further, we briefly outline the learning automata techniques based on which our threshold optimization algorithm is proposed.

A. Learning Automata

Learning automata techniques are first introduced in the control community where in many scenarios, the system is time-varying and stochastic in nature. Therefore, stochastic learning approaches are desired to address the stochastic control problem in random systems. The basic idea of learning automata techniques can be described as follows. We consider a stochastic environment and a set of finite actions available for the decision maker. Each selected action induces an output from the random environment. However, due to the stochastic nature, the outputs for a given input may be different at different time instances. Based on observations, a learning automata algorithm is expected to find an action which is the stochastic optimum solution in the sense that the expected system's objective is maximized.

At each decision instance, the decision maker selects one of the actions according to a probability vector. The selected action is fed to the stochastic environment as the input and a random output is attained. The gist of learning automata techniques lies in the provable convergence to the ϵ-optimal solution, as will be defined later, in arbitrary stationary random environment. Thanks to the practicality and applicability, learning automata techniques have been broadly studied in various aspects of the communication and networking literature such as [32] [33] [34] [35]. In this paper, we propose a learning automata based threshold optimization algorithm which finds the stochastic optimum values of the up/down thresholds efficiently in any stationary yet potentially fast-varying random channel environment. The detailed implementations are introduced next.

B. Achieving the Stochastic Optimal Thresholds

We consider an IEEE 802.11 station as the decision maker which adjusts the values of θ_u and θ_d. Without loss of generality, we assume that the maximum value of θ_u and θ_d are m_u and m_d, i.e., the feasible set of θ_u is $\mathcal{A}_u = \{1, \cdots, m_u\}$ and that of θ_d is given by $\mathcal{A}_d = \{1, \cdots, m_d\}$. The station maintains two probability vectors P_u and P_d for $\mathcal{A}_u$ and $\mathcal{A}_d$, respectively. The k-th element in P_u, i.e., P_u^k, denotes the probability that θ_u is set to k. P_d^k is defined analogously. Simply put, at each decision instance, the threshold optimization algorithm randomly determines the values of θ_u and θ_d according to P_u and P_d. Next, the probability vectors, i.e., P_u and P_d, are updated and the iteration continues until convergence, i.e., $P_u^{m^*} = 1$ and $P_d^{n^*} = 1$ where m^* and n^* denote the stochastic optimal values of θ_u and θ_d, respectively.

Define a time series denoted by $\mathbf{t} = [t_0, t_1, \cdots]$ where t_0 denotes the starting time and other elements represent the exact time instances when the data rate is changed. Denote

$$T(j) = [t_{j-1}, t_j], \ j = 1, 2, \cdots$$

as the j-th *time period*, or *time duration*. It is worth noting that within a particular time period, say j, the value of the data rate, the values of thresholds, i.e., θ_u and θ_d , are all fixed numbers. We denote these period-dependent parameters as[3] $r(j)$, $\theta_u(j)$ and $\theta_d(j)$. With this observation, we utilize the time series $\mathbf{t}$ as the time series of decision instances. More specifically, for example, at time t_j, a rate change is triggered either by $\theta_u(j)$ consecutive successful transmissions or $\theta_d(j)$ consecutive failed transmissions. In tandem with the data rate up/down shift, new values of the thresholds, i.e., $\theta_u(j+1)$ and $\theta_d(j+1)$ are determined by the threshold optimization algorithm according to the probability vectors P_u and P_d.

Besides P_u and P_d, we introduce additional vectors for $\mathcal{A}_u$ and $\mathcal{A}_d$, namely, the counting vectors C_u, C_d, the surplus vectors S_u, S_d, the estimation vectors D_u, D_d and the comparison vectors Z_u, Z_d. The definitions of parameters of the algorithm are provided as follows.

Algorithm:

Parameters[4]:

- P_u (P_d): The probability vector for θ_u (θ_d) over $\mathcal{A}_u$ ($\mathcal{A}_d$).
- m_u (m_d): The maximum value of θ_u (θ_d), or equivalently, the cardinality of $\mathcal{A}_u$ ($\mathcal{A}_d$).
- C_u (C_d): The counting vector of θ_u (θ_d) where the k-th element, i.e., C_u^k (C_d^k), denotes the times that k has been selected as the value of θ_u (θ_d).
- S_u (S_d): The surplus vector of θ_u (θ_d) where the k-th element, i.e., S_u^k (S_d^k), denotes the accumulated throughput with $\theta_u = k$ ($\theta_d = k$).
- D_u (D_d): The estimation vector of θ_u (θ_d) where the k-th element, i.e., D_u^k (D_d^k), is calculated by $D_u^k = \frac{S_u^k}{C_u^k}$ ($D_d^k = \frac{S_d^k}{C_d^k}$).
- R: The resolution parameter which is a positive integer and is tunable by the station.
- δ_u (δ_d): The stepsize parameter of θ_u (θ_d) and is given by $\delta_u = \frac{1}{m_u \times R}$ ($\delta_d = \frac{1}{m_d \times R}$).
- φ_u (φ_d): The perturbation vector of θ_u (θ_d) where the k-th element, i.e., φ_u^k (φ_d^k), is a zero mean random variable which is uniformly distributed in $[-\frac{\rho}{C_u^k(C_d^k)}, +\frac{\rho}{C_u^k(C_d^k)}]_{-1}^{+1}$ where ρ is a system parameter and is controllable by the station. The notation of $[a,b]_y^x$ represents $[\max(a,y), \min(b,x)]$.
- Z_u (Z_d): The comparison vector of θ_u (θ_d) where the k-th element, i.e., Z_u^k (Z_d^k), is given by $Z_u^k = D_u^k + \varphi_u^k$ ($Z_d^k = D_d^k + \varphi_d^k$).
- B: The predefined convergence threshold, e.g., 1, which is determined by the station.
- J: A running parameter which records the updated maximum achieved throughput during one time period.

[3]Note that with a slight abuse of notation, we use j to denote the j-th time duration.

[4]We present the analogous definitions in the parenthesis.

At time t_0:

Initialization:

- The station sets $P_u = [p_1, \cdots, p_i, \cdots, p_{m_u}]$ where $p_i = \frac{1}{m_u}$ for all $1 \leq i \leq m_u$. Similarly, P_d is given by $[p_1, \cdots, p_i, \cdots, p_{m_d}]$ where $p_i = \frac{1}{m_d}$ for all $1 \leq i \leq m_d$.
- Initializes C_u, C_d, S_u, S_d, D_u and D_d to zeros.
- Randomly selects the values of $\theta_u(1)$ and $\theta_d(1)$, say m, n, according to $P_u(1)$ and $P_d(1)$.
- Transmits with $\theta_u = m$ and $\theta_d = n$ until $T(1)$ ends, i.e., a data rate change is triggered.
- Records the average throughput within $T(1)$ as J.

At time $t_j (j \geq 1)$:

Do:

- Records the average throughput during T_j as $O(j)$. If $O(j) > J$, sets $J = O(j)$ and remains J unchanged otherwise.
- Updates the m-th element in the surplus vector S_u and the n-th element in S_d by adding the measured normalized throughput of the last time period, as $S_u^m = S_u^m + \frac{O(j)}{J}$ and $S_d^n = S_d^n + \frac{O(j)}{J}$.
- Updates the counting vectors by adding one to the m-th counter in C_u and the n-th counter in C_d, as $C_u^m = C_u^m + 1$ and $C_d^n = C_d^n + 1$.
- Updates the m-th element in D_u and the n-th element in D_d by $D_u^m = \frac{S_u^m}{C_u^m}$ and $D_d^n = \frac{S_d^n}{C_d^n}$.
- For every element in Z_u and Z_d, updates $Z_u^k = D_u^k + \varphi_u^k,\ k = 1, \cdots, m_u$ and $Z_d^k = D_d^k + \varphi_d^k,\ k = 1, \cdots, m_d$.
- Finds the element in Z_u with the highest[5] value of $Z_u^k, k = 1, \cdots, m_u$, say, the $\tilde{m}$-th element in $\mathcal{A}_u$.
- Similarly, finds the element in Z_d which has the highest $Z_d^k, k = 1, \cdots, m_d$, say, the $\tilde{n}$-th element in $\mathcal{A}_d$.
- Updates the probability vectors of P_u and P_d as

$$\begin{aligned} P_u^k &= \max\left(P_u^k - \delta_u, 0\right) \quad \text{if } k \neq \tilde{m}, k = 1, \cdots, m_u \\ P_u^k &= 1 - \sum_{k \neq \tilde{m}} P_u^k \text{ if } k = \tilde{m} \end{aligned} \tag{24}$$

and

$$\begin{aligned} P_d^k &= \max\left(P_d^k - \delta_d, 0\right) \quad \text{if } k \neq \tilde{n}, k = 1, \cdots, m_d \\ P_d^k &= 1 - \sum_{k \neq \tilde{n}} P_d^k \text{ if } k = \tilde{n} \end{aligned} \tag{25}$$

where $\delta_u = \frac{1}{m_u \times R}$ and $\delta_d = \frac{1}{m_d \times R}$.
- With the updated probability vectors P_u and P_d, new values of θ_u and θ_d, i.e., $\theta_u(j+1)$ and $\theta_d(j+1)$, are selected.
- Starts the transmissions in $T(j+1)$ with $\theta_u(j+1)$ and $\theta_d(j+1)$.

Until:

- $\max(P_u) \geq B$ and $\max(P_d) \geq B$ where B is the predefined convergence threshold.

[5] Note that a tie can be easily broken by a random selection.

End

The proposed threshold optimization algorithm is similar to the *stochastic estimator learning automata* proposed in [36]. The key feature of this genre of learning automata is the randomness deliberately introduced by φ_u^k and φ_d^k. Note that although φ_u^k and φ_d^k are zero mean random variables, their variances are dependent on the values of C_u^k and C_d^k, respectively. Specifically, the variances approach to zeros with the increase of the number of times that the corresponding values of θ_u and θ_d are selected. As a consequence, the threshold optimization algorithm inclines to more reliable stochastic estimates and thus possesses a faster convergence behavior than other learning algorithms [36]. The values that have been selected less frequently still have the chance of being considered as optimal. However, the missing probability diminishes to zero along iterations. In the algorithm, the resolution parameter R controls the stepsize of probability adjustment in the algorithm. A smaller value of R produces a fine-grained probability adjustment yet unavoidably prolongs the convergence time. The convergence threshold B determines the stopping criteria of the algorithm. Therefore, a tradeoff between optimality and convergence rate can be adjusted by tuning the value of B.

The steady state behavior of the proposed threshold optimization algorithm is provided in the following theorem.

Theorem 2: The proposed threshold optimization algorithm is *ϵ-optimal* for any stationary channel environment with arbitrary distribution. Mathematically, for any arbitrarily small $\epsilon > 0$ and $\gamma > 0$, there exists a t' satisfying

$$\Pr\{|1 - P_u^{m^*}| < \epsilon\} > 1 - \gamma \ \forall t > t' \tag{26}$$

and

$$\Pr\{|1 - P_d^{n^*}| < \epsilon\} > 1 - \gamma \ \forall t > t' \tag{27}$$

where m^* and n^* are the stochastic optimal values of θ_u and θ_d, respectively.

The proof of Theorem 2 follows similar lines as in [36] and is omitted for brevity. For more discussions on the stochastic estimator algorithms, refer to [37] and [38]. In the next section, we will demonstrate the efficacy of our proposed threshold optimization algorithm via simulations.

V. Performance Evaluation

In this section, we evaluate the performance of the proposed threshold optimization algorithm with simulations. For comparisons, we first implement the heuristics-based threshold adjustment algorithms in [2] and [3]. In [2], the downshift threshold θ_d is fixed to 1 and the default value of θ_u is 10. After θ_u successful transmissions, a data upshift is triggered and if the first transmission after the rate upshift is successful, the algorithm assumes that the link quality is improving rapidly [2]. Consequently, θ_u is set to a small number, e.g., $\theta_u = 3$, in order to capture the fast improving channel. Otherwise, the algorithm assumes that the channel is changing slowly and thus a larger value of θ_u is desired, e.g., $\theta_u = 10$. We denote this threshold adaptation scheme as DLA in our simulations. Another well-known threshold adjusting scheme, namely, AARF, is proposed in [3]. Similarly, the rate downshift threshold is fixed to $\theta_d = 2$ empirically. However, for θ_u, a binary exponential backoff scheme is applied. If the first transmission after a rate upshift failed, the data rate is switched back to the previous rate and the value of θ_u is doubled with a maximum value of 50. The value of θ_u is reset to 10 whenever a rate downshift is triggered.

To simulate the indoor office environment for IEEE 802.11 WLANs, we simulate a Rayleigh fading channel environment. In other words, we assume a flat fading environment. However, it could be either a fast fading or slow fading channel. The Doppler spread (in Hz) corresponds to the channel fading speed where a large Doppler spread value represents a fast fading channel and a small Doppler spread indicates a slow-varying channel. It should be noted that the Doppler spread value describes the time dispersive nature of the wireless channel [39], which is inversely proportional to the channel coherence time. More specifically, we relate the channel coherence time T_c and the Doppler spread value f_m as [39]

$$T_c = \frac{0.423}{f_m}. \tag{28}$$

Therefore, a larger Doppler spread value indicates a small channel coherence time which represents a fast fading scenario. We conduct the simulations under various channel fading conditions via different Doppler spread values. We consider the IEEE 802.11b PHY specification, i.e., the available data rates are 1 Mbps, 2 Mbps, 5.5 Mbps and 11 Mbps and the RTS/CTS signalling scheme is turned off. Since the objective of our proposed threshold optimization algorithm is to combat with the channel variation, the collision effect is omitted. Therefore, we consider a WLAN with one station consistently transmitting packets to the AP. However, note that [21] provides a complementary solution to mitigate the collision effect and thus our algorithms can work collectively as a joint solution. We emphasize that this simplification does not induce any loss of generality since although seemingly simple, it produces all the challenging problems involved in rate adaptation algorithms in a time-varying stochastic channel environment. The data traffic is generated using constant bit rate UDP traffic sources and the frame size is set to 1024 octets. The power of the transmitter and the power of thermal noise are set to 50 mW and 1 mW, respectively. The SNR-BER relation is given by Table 1 which is derived from [40].

We vary the Doppler spread value to simulate the various channel fading speeds. For each value of Doppler spread value, the simulation is executed for 1000 seconds and the average throughput of the following algorithms[6], which are commonly based on the threshold-based up/down mechanism, are compared: (1), *OP* - the optimum throughput attained by assuming an oracle which foresees the variation of channel and adapts the data rate optimally. This curve is attained as a performance comparison benchmark; (2), *ARF* - the threshold-based rate adaptation algorithm with $\theta_u = 10$ and $\theta_d = 2$; (3), *DLA* - the dynamic threshold adjustment algorithm proposed in [2]; (4), *AARF* - the threshold adjustment algorithm proposed in [3]; (5), *TOA* - the threshold optimization algorithm proposed in this paper. The algorithms are compared with each other in terms of the average system throughput (in Mbps). For TOA, without loss of generality, we assume that $m_u = m_d = 10$ and the resolution parameter R is 1. The convergence threshold B is 0.999 and ρ is 1. The performance curves of the aforementioned algorithms are plotted in Fig. 1.

[6]The performance comparison of ARF and other open-loop rate adaptation algorithms such as SampleRate and ONOE has been studied extensively in [5] and [7] for various channel conditions.

TABLE I
SNR V.S. BER FOR IEEE 802.11b DATA RATES

SNR (dB)	BPSK (1Mbps)	QPSK (2Mbps)	CCK (5.5 Mbps)	CCK (11 Mbps)
1	$1.2E^{-5}$	$5E^{-3}$	$8E^{-2}$	$1E^{-1}$
2	$1E^{-6}$	$1.2E^{-3}$	$4E^{-2}$	$1E^{-1}$
3	$6E^{-8}$	$2.1E^{-4}$	$1.8E^{-2}$	$1E^{-1}$
4	$7E^{-9}$	$3E^{-5}$	$7E^{-3}$	$5E^{-2}$
5	$2.3E^{-10}$	$2.1E^{-6}$	$1.2E^{-3}$	$1.3E^{-2}$
6	$2.3E^{-10}$	$1.5E^{-7}$	$3E^{-4}$	$5.2E^{-3}$
7	$2.3E^{-10}$	$1E^{-8}$	$6E^{-5}$	$2E^{-3}$
8	$2.3E^{-10}$	$1.2E^{-9}$	$1.3E^{-5}$	$7E^{-4}$
9	$2.3E^{-10}$	$1.2E^{-9}$	$2.7E^{-6}$	$2.1E^{-4}$
10	$2.3E^{-10}$	$1.2E^{-9}$	$5E^{-7}$	$6E^{-5}$
⋮	⋮	⋮	⋮	⋮

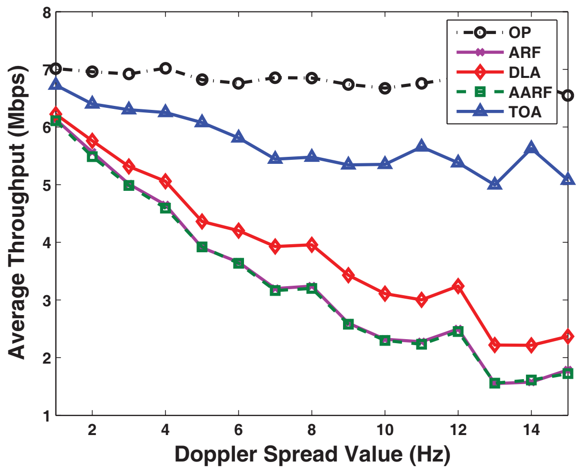

Fig. 1. Average throughput v.s. Doppler spread values.

In Fig. 1, we observe that except the OP curve, all other rate adaptation algorithms suffer from performance degradations with large Doppler spread values. Recall that a large Doppler spread value indicates a fast fading channel environment and hence the average throughput deteriorates due to the incompetency of capturing short channel fluctuations. Among which, ARF and AARF provide worst performance with a slight difference. DLA performs better due to the capability of switching between the large value and the small value of θ_u for different channel conditions. Our proposed threshold optimization algorithm, as demonstrated in Figure 1, consistently outperforms other open-loop rate adaptation algorithms and bridges the performance gap with the OP curve. This superiority becomes remarkably significant in fast fading stochastic channel environments, i.e., scenarios with large Doppler spread values. While other algorithms are jeop-

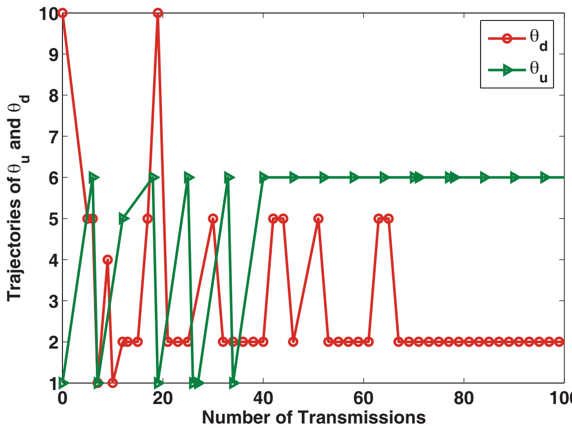

Fig. 2. The trajectories of θ_u and θ_d in achieving the stochastic optimum values.

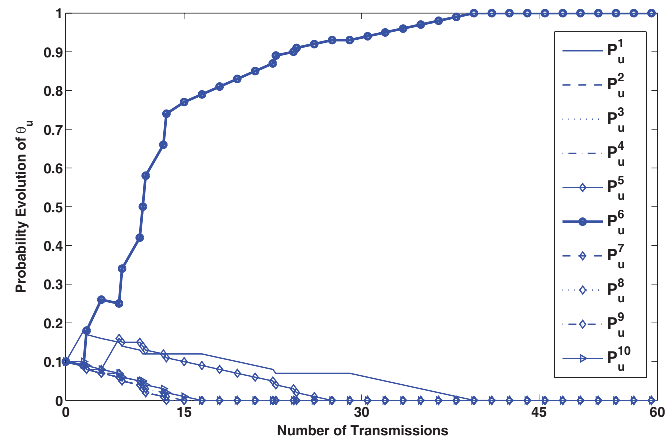

Fig. 3. Evolution of the probability vector P_u.

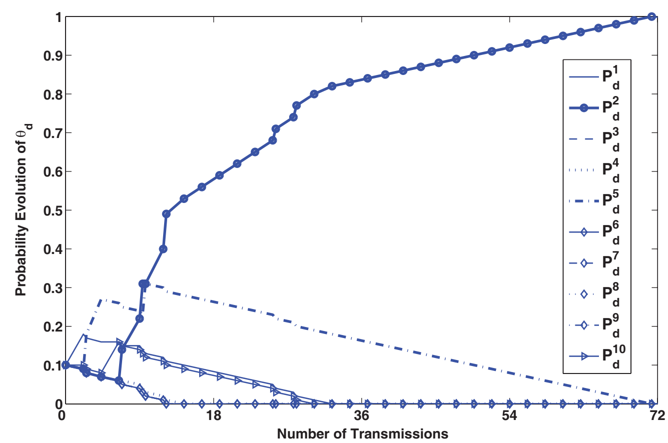

Fig. 4. Evolution of the probability vector P_d.

ardized by the unmature rate changes due to the uncertainty caused by the random and fast-varying channel environment, our proposed threshold optimization algorithm, based on the learning automata techniques, is able to find the stochastic optimum values of θ_u and θ_d which maximize the expected system performance. Therefore, our scheme is particularly suitable for fast changing yet statistically stationary random channel environments where other open-loop rate adaptation solutions usually provide unsatisfactory performance.

To illustrate the process of finding the stochastic optimum values of θ_u and θ_d, in Figure 2, we provide the trajectories of θ_u and θ_d in a sample simulation run with a fixed Doppler spread value of 10. It is observable that starting from the initial point, the threshold optimization algorithm adapts the values of θ_u and θ_d on the fly along with the rate changes. The algorithm soon finds the stochastic optimum values and θ_u and θ_d converge to the optimum solutions effectively. The evolutions of the probability vectors, i.e., P_u and P_d, are demonstrated in Figure 3 and Figure 4, respectively. Note that, in Figure 3, the sixth curve which represents the value of P_u^6 soon excels others and approaches to 1. Correspondingly, the value of θ_u converges to 6 rapidly as depicted in Figure 2. In Figure 4, the second curve approaches to unity gradually while others diminish to zeros. As a consequence, in Figure 2, the value of θ_d converges to the stochastic value, i.e., 2. In this sample run of simulation, the stochastic optimum values of θ_u and θ_d, which maximizes the expected throughput of the system, is given by $\theta_u = 6$ and $\theta_d = 2$. The proposed threshold optimization algorithm finds these values effectively and efficiently, while providing superior performance than other non-adaptive threshold-based rate adaptation algorithms, as demonstrated in Figure 1.

VI. Conclusions and Future Work

In this paper, we investigate the threshold-based rate adaptation algorithm which is predominantly utilized in practical IEEE 802.11 WLANs. Although widely deployed, the obscure objective function of this type of rate adaptation algorithms, commonly based on the heuristic up/down mechanism, is less comprehended. In this work, we study the threshold-based rate adaptation algorithm from a reverse engineering perspective. The implicit objective function, which the rate adaptation algorithm is maximizing, is unveiled. Our results provide, albeit approximate, an analytical model from which the threshold-based rate adaptation algorithm, such as ARF, can be better understood.

In addition, we propose a threshold optimization algorithm which dynamically adapts the up/down thresholds, based on the learning automata techniques. Our algorithm provably converges to the stochastic optimum solutions of the thresholds in arbitrary stationary yet potentially fast changing random channel environment. Therefore, by combining our work with the threshold adaptation scheme in [21], where the thresholds are adjusted to mitigate the collision effects, a joint collision and random channel fading resilient solution can be attained.

In this paper, to emphasize on the impact of random channel variations, we restrict ourselves to the scenario where all stations have a fixed and known transmission probability p. The interaction of the transmission probabilities and the data rates seems interesting and needs further investigation. Due to the competitive nature of channel access, a stochastic game formulation may be utilized. Moreover, our work assumes a fixed length L for every packet. A natural extension to the joint threshold and frame size optimization is our on-going research following [41] and [42].

REFERENCES

[1] D. Qiao and S. Choi, "Fast-responsive link adaptation for IEEE 802.11 WLANs," in *Proc. IEEE ICC*, 2003.
[2] P. Chevillat, J. Jelitto, A. N. Batteto, and H. L. Truong, "A dynamic link adaptation algorithm for IEEE 802.11a wireless LANs," in *Proc. IEEE ICC*, 2003.
[3] M. Lacage, M. H. Manshaei, and T. Turletti, "IEEE 802.11 rate adaptation: a practical approach," *ACM MSWiM*, 2004.
[4] J. Kim, S. Kim, S. Choi, and D. Qiao, "CARA: collision-aware rate adaptation for IEEE 802.11 WLANs," in *Proc. IEEE Infocom*, 2006.
[5] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan, "Robust rate adaptation for 802.11 wireless networks," in *Proc. ACM MobiCom*, 2006.
[6] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, "Opportunistic media access for multirate ad hoc networks," in *Proc. ACM MobiCom*, 2002.
[7] S. Pal, S. R. Kundu, K. Basu, and S. K. Das, "IEEE 802.11 rate control algorithms: Experimentation and performance evaluation in infrastructure mode," the University of Texas at Arlington, Tech. Rep., 2006.
[8] Q. Pang, V. C. M. Leung, and S. C. Liew, "A rate adaptation algorithm for IEEE 802.11 WLANs based on MAC-layer loss differentiation," in *Proc. IEEE BroadNets*, 2005.
[9] H. Jung, K. Cho, Y. Seok, T. Kwon, and Y. Choi, "RARA: rate adaptation using rate-adaptive acknowledgement for IEEE 802.11 WLANs," in *Proc. IEEE CCNC*, 2008.
[10] G. Judd, X. Wang, and P. Steenkiste, "Low-overhead channel-aware rate adaptation," *ACM MobiCom (extended abstract)*, 2007.
[11] J. C. Bicket, "Bit-rate selection in wireless networks," Master's thesis, Massachusetts Institute of Technology, 2005.
[12] G. Holland, N. Vaidya, and P. Bahl, "A rate-adaptive mac protocol for multi-hop wireless networks," in *Proc. ACM MobiCom*, 2001.
[13] K. Ad and M. Leo, "WaveLAN(C)-II: a high-performance wireless LAN for the unlicensed band," *Bell Labs Techn. J.*, vol. 2, pp. 118–133, 1997.
[14] I. Haratcherev, K. Langendoen, R. Lagendijk, and H. Sips, "Hybrid rate control for IEEE 802.11," in *Proc. ACM MobiWac*, 2004.
[15] [Online]. Available: http://madwifi.org/browser/madwifi/trunk/ath_rate/onoe
[16] J.-H. Yun and S.-W. Seo, "Novel collision detection scheme and its applications for IEEE 802.11 wireless LANs," *Computer Commun.*, vol. 30, 2007.
[17] S.-C. Wang and A. Helmy, "BEWARE: background traffic-aware rate adaptation for IEEE 802.11 MAC," in *Proc. IEEE WoWMoM*, 2008.
[18] [Online]. Available: http://secure.enterasys.com/products/wireless/RBTBG-AW
[19] J. Choi, K. Park, and C. kwon Kim, "Cross-layer analysis of rate adaptation, DCF and TCP in multi-rate WLANs," in *Proc. IEEE Infocom*, 2007.
[20] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee, "Diagnosing wireless packet losses in 802.11: separating collision from weak signal," in *Proc. IEEE Infocom*, 2008.
[21] J. Choi, J. Na, K. Park, and C. kwon Kim, "Adaptive optimization of rate adaptation algorithms in multi-rate WLANs," in *Proc. IEEE ICNP*, 2007.
[22] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *J. Operation Research Society*, vol. 49, pp. 237–252, 1998.
[23] S. H. Low and D. E. Lapsley, "Optimization flow control–I: basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–874, 1999.
[24] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: a mathematical theory of network architectures," *Proc. IEEE*, vol. 95, pp. 255–312, Mar. 2007.
[25] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The stable paths problem and interdomain routing," *IEEE/ACM Trans. Networking*, vol. 10, pp. 232–243, 2002.
[26] J.-W. Lee, A. Tang, J. Huang, M. Chiang, and A. R. Calderbank, "Reverse-engineering MAC: a non-cooperative game model," *IEEE J. Sel. Areas Commun.*, June 2007.
[27] F. Cali, M. Conti, and E. Gregori, "Dynamic tuning of the IEEE 802.11 protocol to achieve a theoretical throughput limit," *IEEE/ACM Trans. Networking*, vol. 8, 2000.
[28] P. Glasserman, *Gradient Estimation via Perturbation Analysis*. Springer, 1991.
[29] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems (Second Edition)*. Springer, 2008.
[30] X.-R. Cao, "A basic formula for online policy gradient algorithms," *IEEE Trans. Automatic Control*, vol. 50, 2005.
[31] ——, *Stochastic Learning and Optimization: A Sensitivity-Based Approach*. Springer, 2007.
[32] Y. Xing and R. Chandramouli, "Price dynamics in competitive agile spectrum access markets," *IEEE J. Sel. Areas Commun.*, Mar. 2007.
[33] M. A. Haleem and R. Chandramouli, "Adaptive downlink scheduling and rate selection: a cross layer design," *IEEE J. Sel. Areas Commun.*, vol. 23, June 2005.
[34] S. Kiran and R. Chandramouli, "An adaptive energy-efficient link layer protocol using stochastic learning control," in *Proc. IEEE ICC*, 2003.
[35] Y. Xing and R.Chandramouli, "Distributed discrete power control for bursty transmissions over wireless data networks," in *Proc. IEEE ICC*, 2004.
[36] G. I. Papadimitriou, M. Sklira, and A. S. Pomportsis, "A new class of ϵ-optimal learning automata," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 34, pp. 346–354, 2004.
[37] M. A. L. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 32, Dec. 2002.
[38] ——, *Networks of Learning Automata - Techniques for Online Stochastic Optimization*. Kluwer Academic Publishers, 2004.
[39] T. S. Rappaport, *Wireless Communications Principles and Practices*. Prentice Hal, 2002.
[40] J. del Prado Pavon and S. Choi, "Link adaptation strategy for IEEE 802.11 WLAN via received signal strength measurement," in *Proc. IEEE ICC*, 2003.
[41] C.-C. Chen, E. Seo, H. Luo, N. H. Vaidya, and X. Wang, "Rate-adaptive framing for interfered wireless networks," in *Proc. IEEE Infocom*, 2007.
[42] D. Qiao and S. Choi, "Goodput enhancement of IEEE 802.11a wireless LAN via link adaptation," in *Proc. IEEE ICC*, 2001.

Yang Song received his B.E. and M.E. degrees in Electrical Engineering from Dalian University of Technology, Dalian, China, and University of Hawaii at Manoa, Honolulu, U.S.A., in July 2004 and August 2006, respectively. Since September 2006, he has been working towards the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, Florida, USA. His research interests are wireless networks, algorithmic game theory, network economics, and stochastic network optimization. He is a student member of IEEE, a student member of ACM, and a member of Game Theory Society.

Xiaoyan Zhu received her BE degree in Information Engineering from Xidian University, Xian, China, in July 2000, and her ME degree in Information and Communications Engineering from Xidian University, Xian, China, in March 2004. She is now working towards her Ph.D. degree at Xidian University and is currently a visiting research scholar with Wireless Networks Laboratory (WINET) in the Department of Electrical and Computer Engineering at University of Florida. Her research interests include wireless networks, network security and network coding.

Yuguang Fang (biography and photo were not available at the time of publication.)

Hailin Zhang (biography and photo were not available at the time of publication.)