Analytical Modeling of TCP Dynamics in Infrastructure-Based IEEE 802.11 WLANs

Jeonggyun Yu, Sunghyun Choi, and Daji Qiao

Abstract: **IEEE 802.11 wireless local area network (WLAN) has become the prevailing solution for wireless Internet access while transport control protocol (TCP) is the dominant transport-layer protocol in the Internet. It is known that, in an infrastructure-based WLAN with multiple stations carrying *long-lived* TCP flows, the number of TCP stations that are actively contending to access the wireless channel remains very small. Hence, the aggregate TCP throughput is basically independent of the total number of TCP stations. This phenomenon is due to the closed-loop nature of TCP flow control and the bottleneck downlink (i.e., access point-to-station) transmissions in infrastructure-based WLANs. In this paper, we develop a comprehensive analytical model to study TCP dynamics in infrastructure-based 802.11 WLANs. We calculate the average number of active TCP stations and the aggregate TCP throughput using our model for given total number of TCP stations and the maximum TCP receive window size. We find out that the default minimum contention window sizes specified in the standards (i.e., 31 and 15 for 802.11b and 802.11a, respectively) are not optimal in terms of TCP throughput maximization. Via ns-2 simulation, we verify the correctness of our analytical model and study the effects of some of the simplifying assumptions employed in the model. Simulation results show that our model is reasonably accurate, particularly when the wireline delay is small and/or the packet loss rate is low.**

Index Terms: **Distributed coordination function (DCF), IEEE 802.11 wireless local wirelss network (WLAN), infrastructure-based, transport control protocol (TCP) dynamics.**

I. INTRODUCTION

In recent years, IEEE 802.11 wireless local wirelss network (WLAN) has gained a prevailing position in the market for (indoor) broadband wireless networking. The 802.11 standard defines both the medium access control (MAC) layer and the physical layer (PHY) specifications [1]. The mandatory part of the 802.11 MAC is called the distributed coordination function (DCF), which is based on carrier sense multiple access with collision avoidance (CSMA/CA).

As pointed out in [2], more than 90% of the Internet traffic is carried by the transport control protocol (TCP). Consequently, a lot of data traffic over the 802.11 WLAN is TCP traffic. Therefore, it is critical to have a good understanding of the TCP dynamics over WLAN. It has been shown in [3], [4] that, with multiple stations carrying *long-lived* TCP flows in an infrastructure-based 802.11 WLAN, the number of stations that are actively contending remains very small. Accordingly, the aggregate TCP throughput is almost independent of the total number of TCP stations in the network. This interesting phenomenon is due to (i) the closed-loop nature of TCP flow control and (ii) the bottleneck downlink (i.e., access point (AP)-to-station) transmissions in the infrastructure-based WLANs. More specifically, since a long-lived upload TCP flow typically operates in the congestion avoidance phase, the source station can transmit a TCP Data only when it receives a TCP acknowledgement (ACK) from its AP. In the meanwhile, each download TCP station only replies with a TCP ACK upon reception of a TCP data from the AP. On the other hand, the 802.11 DCF guarantees long-term equal channel access probabilities among contending stations in the network, regardless whether it is an AP or a station. As a result, most of the outstanding TCP packets (in the form of either TCP ACK for upload or TCP data for download) are accumulated at the AP, while most of the stations remain *inactive* waiting for incoming packets from the AP.

There have been efforts to understand the TCP dynamics over 802.11 WLANs [4], [5]–[8]. In [4], Choi *et al.* analyzed the number of active stations using a Markov chain, but they simply computed the state transition probabilities based on empirical results. In [5], Kherani *et al.* conducted a theoretical analysis on the TCP performance over multi-hop 802.11 WLANs; in contrast, our study focuses on infrastructure-based WLANs. Burmeister *et al.* analyzed the TCP over multi-rate WLANs [6]. However, they approximated the AP behavior as an M/G/1/B queuing system and did not study the effects of contention-based MAC on TCP dynamics in detail. In [7], Choi *et al.* analyzed the TCP performance over 802.11 WLANs by using an approximated birth-death chain and a fixed-point technique. Our study provides more comprehensive details to understand the impact of DCF MAC on the TCP dynamics. Moreover, we investigate the impact of various parameters such as maximum TCP receive window size and contention window size on the TCP performance. Bruno *et al.* approached this problem using a p-persistent DCF model [8] with a critical assumption that no more than one TCP packet is enqueued in the buffer of an active station. This implies that a station becomes inactive after transmitting its packet successfully while each successful transmission from the AP always activates a different station, which is not true in practice. In a real network, the number of outstanding packets in a long-lived TCP flow is usually larger than one.

Manuscript received March 18, 2008; approved for publication by Young Han Kim, Division III Editor, June 05, 2009.

This work was in part supported by MKE/IITA, Korea under both the IT R&D program [2008-F-007-01, Intelligent Wireless Communication Systems in 3 Dimensional Environment] and the Information Technology Research Center support program [IITA-2009-C1090-0902-0006].

J. Yu is with Samsung Electronics Co., LTD, Suwon, Korea, email: jgyu@mwnl.snu.ac.kr. This work was done while he was a Ph.D. student at Seoul National University.

S. Choi is with the School of Electrical Engineering and INMC, Seoul National University, Seoul 151-744, Korea, email: schoi@snu.ac.kr.

D. Qiao is with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, USA, email: daji@iastate.edu.

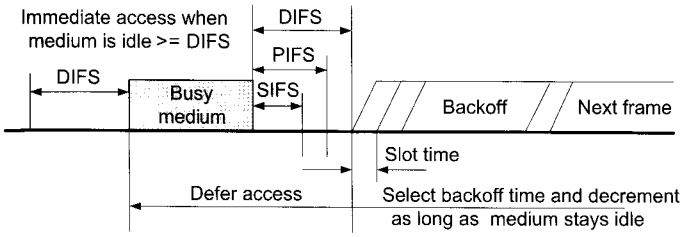


Fig. 1. IEEE 802.11 DCF channel access scheme.

Moreover, each successful transmission from the AP may not always activate a different station, because the AP might transmit a packet to an already active station.

In this paper, we conduct a rigorous and comprehensive analysis of TCP dynamics over DCF-based 802.11 WLANs. Our analytical model is based on the p-persistent model developed by Cali *et al.* [9]. We calculate the average number of actively contending TCP stations and the aggregate TCP throughput using our analytical model. Then, we verify the analytical results by comparing them with ns-2 simulation results. Our analysis also shows that the default minimum contention window sizes specified in the current standards (i.e., 31 and 15 for 802.11b and 802.11a, respectively) are too large to achieve the maximum aggregate TCP throughput.

The rest of the paper is organized as follows. In Section II, we briefly discuss the 802.11 DCF, the p-persistent CSMA/CA model, and the dynamics of long-lived TCP flows. Section III describes the details of our analytical model for TCP dynamics over the 802.11 DCF. The model is evaluated via simulation study in Section IV. Finally, we conclude the paper with discussion of future work in Section V.

II. PRELIMINARIES

A. IEEE 802.11 DCF

IEEE 802.11 MAC [1] defines two coordination functions, namely, the mandatory DCF based on CSMA/CA and the optional point coordination function (PCF) based on a poll-and-response mechanism. Most of today's 802.11 devices operate only in the DCF mode. Next, we briefly explain how the DCF works since our model is for the DCF-based WLAN.

The CSMA/CA of the DCF works as follows. When a packet arrives at the head of a transmission queue, if the channel is busy, the MAC waits until the medium becomes idle, then defers for an extra time interval, called DCF inter-frame space (DIFS). If the channel stays idle during the DIFS deference, the MAC starts the backoff process by selecting a random backoff counter. For each idle slot time elapsed, the backoff counter is decremented by one. When the counter reaches zero, the packet is transmitted. The timing of the DCF channel access is illustrated in Fig. 1.

Each station maintains a contention window (CW), which is used to select the random backoff counter. The backoff counter is determined as a random integer drawn from a uniform distribution over the interval [0, CW]. If the channel becomes busy during a backoff process, the backoff is suspended. When the channel becomes idle again and stays idle for DIFS time, the backoff process resumes from the suspended backoff counter value. For each successful reception of a packet, the receiving station immediately acknowledges by sending an ACK packet. The ACK packet is transmitted after a short inter-frame space (SIFS), which is shorter than DIFS. If an ACK packet is not received after the data transmission, the packet is retransmitted after another random backoff. The CW size is initially set to CWmin, and increases to $2 \times (\mathrm{CW}+1)-1$ upon each transmission failure with an upper bound of CWmax.

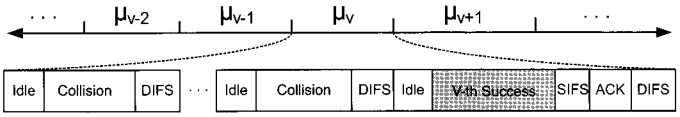


Fig. 2. Illustration of virtual slots (μ) in the p-persistent model.

In an 802.11 DCF WLAN, after a station transmits a packet successfully, it resets its CW value to CWmin, and performs DIFS deference and random backoff. This is often referred to as *post backoff* since this backoff is done after, not before, a transmission. Post backoff ensures that there is at least one backoff interval between two consecutive frame transmissions. On the other hand, when there is no on-going backoff and when the channel has been idle for longer than DIFS time, a frame can be transmitted immediately without additional backoff. This is referred to as *immediate access*.

B. p-persistent CSMA/CA Model

Cali *et al.* [9] studied the DCF behavior with a p-persistent model. Instead of using the binary exponential backoff, the p-persistent model determines the backoff interval by sampling from a geometric distribution with parameter p. Thanks to the memoryless property of the geometric distribution, it is more tractable to analyze the p-persistent model. Based on the geometric backoff assumption, the processes that define the occupancy behavior of the channel (i.e., idle slots, collisions, and successful transmissions) are regenerative, where the regenerative points correspond to completions of successful transmissions. As shown in Fig. 2, the vth virtual slot, denoted by μ_v, consists of the vth successful transmission and its preceding idle periods and collision periods. An idle period is a time interval during which all the backlogged stations are performing backoff without transmission attempts. A collision period is an interval during which two or more stations transmit simultaneously and collide with each other.

Statistically, the DCF operation and its p-persistent model are equivalent when the stations are always active (i.e., when they always have a frame to transmit). However, when a station carries long-lived TCP flows, it may not always have a frame to transmit. For example, during the congestion avoidance phase, such a station is allowed to transmit an additional TCP data only after it receives a TCP ACK for one of its outstanding TCP data packets. Therefore, when analyzing the TCP dynamics over WLAN, there are slight differences between the standard DCF operation and its corresponding p-persistent model. Nevertheless, our analysis is based on the p-persistent model for simplicity, and the accuracy of the analysis will be evaluated in Section IV.

C. Dynamics of Long-Lived TCP Flows

Though there are different versions of TCP protocols, a TCP flow always operates in either slow-start or congestion avoidance phase regardless of the TCP version. The slow-start phase occurs during startup as well as when a timeout occurs at the sender due to packet loss. In this paper, we consider long-lived TCP flows so that we may ignore the effects of slow start since its fraction in the flow lifetime is very small.

In an early version of TCP, known as *TCP Tahoe*, the sender resets its congestion window to one maximum segment size (MSS) and enters the slow start phase after each packet loss. This deficiency was remedied in later TCP versions such as *TCP NewReno*, which has been implemented in Microsoft Windows XP. The key difference between TCP NewReno and TCP Tahoe is the addition of *Fast Retransmit* and *Fast Recovery* mechanisms [10]. Fast Retransmit prevents the timeout by retransmitting a packet when three duplicate ACKs are received at the sender. Fast Recovery cancels the slow start phase by setting the congestion window to approximately half of its current value and keeping the connection in the congestion avoidance phase. Although bursts of packet losses may still cause timeouts at the sender and subsequently force slow start to be invoked, TCP NewReno recovers from slow start much faster than earlier versions of TCP [11].

Because the current 802.11 WLAN has a smaller bandwidth than the wireline network, WLAN is very likely to be the bottleneck in the round trip path of TCP traffic. Therefore, most of the outstanding TCP packets stay in WLAN for most of their life time. Moreover, in the absence of frequent or bursty packet losses in the wireline network, the TCP send window may grow up to the maximum window size allowed by the receiver, i.e., the maximum TCP receive window size.

III. ANALYTICAL MODEL FOR TCP DYNAMICS OVER 802.11 WLANS

In this section, we analyze the average number of active TCP stations and the aggregate TCP throughput in an 802.11 WLAN under the assumption of ideal channel conditions (i.e., no hidden terminals, no channel error, and no capture effect).

A. Network Model and Assumptions

We consider an infrastructure-based 802.11 WLAN with a single AP and N_{STA} stations; each station carries long-lived TCP flows. Stations either download from or upload data to remote FTP servers. We conduct our analysis under the following assumptions for simplicity and the effects of some of the assumptions will be studied in Section IV.

[A1] The queue sizes of the AP and stations are large enough to avoid buffer overflow. We will investigate the effects of the queue sizes on the aggregate TCP throughput in Section IV-E.

[A2] The successful transmission of each TCP data is notified by an immediate TCP ACK instead of a delayed TCP ACK. The effects of delayed TCP ACK scheme on the aggregate TCP throughput will be studied in Section IV-D.

Table 1. List of notations and symbols.

Term	Definition
W	Maximum TCP receive window size (in packets)
M	Total # of states
N_{STA}	Total # of TCP stations in an infrastructure-based WLAN
$\vec{N}_k$	*State* k vector
N_k^i	ith element of the *State* k vector
η_k	Number of active TCP stations in *State* k
$P_{m,k}$	State transition probability from *State* m to *State* k
Q_k^{AP}	Length of the AP queue in *State* k
$P_{\text{STA},k}^{\text{succ}}$	Probability that a non-AP station succeeds in a transmission attempt in *State* k
$P_{\text{AP},k}^{\text{succ}}$	Probability that the AP succeeds in a transmission attempt in *State* k
$P_{\text{AP},k}^i$	Probability that the AP succeeds in transmitting a packet to a class-i station in *State* k.
$P_{\text{STA},k}^i$	Probability that a class-i station succeeds in transmitting a packet in *State* k

[A3] As mentioned in Section II-C, long-lived TCP flows often operate in the congestion avoidance phase during most of their lifetime, as long as packet loss does not occur in bursts frequently. In our model, we assume ideal channel conditions (i.e., no hidden terminals, no channel error, and no capture effect). We will study the effects of packet loss on the aggregate TCP throughput in Section IV-G.

[A4] In practice, the maximum TCP receive window size is often a small number, typically smaller than the TCP congestion window size during the congestion avoidance phase. We also know that the number of outstanding packets is determined by the minimum of the TCP congestion window size and the TCP receive window size. Based on this observation and [A3], we assume that the number of outstanding packets (including TCP data and TCP ACK) for a TCP flow is equal to the maximum TCP receive window size (in packets).[1]

[A5] There is no TCP ACK processing delay between a TCP data reception and the corresponding TCP ACK transmission. We will study the effects of TCP ACK processing delay on the aggregate TCP throughput in Section IV-F.

B. State Space

Table 1 lists the notations and symbols used in our analysis. We use a discrete-time Markov chain to model the distribution of the number of stations carrying different numbers of TCP packets in their queues at the end of the vth virtual slot in the p-persistent CSMA/CA model.

$\vec{N}(v) = (N^0(v), N^1(v), N^2(v), \cdots, N^W(v))$ is the state vector, where $N^i(v)$ represents the number of stations, each of which carries i TCP packets in its queue at the end of the vth virtual slot. We refer to such stations as *class-i* stations. Therefore, $N^0(v)$ is the number of stations with no TCP packets in their

[1]The TCP window sizes are actually maintained in bytes. In our model, for simplicity, we assume that the TCP data packets have a fixed size and are transmitted at a fixed rate. Therefore, the TCP window sizes can be measured in packets. Similar analysis could be done for variable packet sizes and multi-rate WLANs, which is omitted due to space limitation.

queue. In other words, $N^0(v)$ is the number of inactive stations. W is the maximum TCP receive window size, or equivalently, the number of outstanding TCP packets under assumption [A4]. For a class-i station, i packets reside in its queue while the remaining $(W-i)$ outstanding packets reside in the AP queue.

Our Markov chain is shown in Fig. 3 with the index of each state shown as the number above it. At *State* k, the following equation always holds:

$$N_{\text{STA}} = \sum_{i=0}^{W} N_k^i$$

where N_k^i is the ith element of the *State* k vector, or equivalently, the number of class-i stations at *State* k. Hence N_k^0 is the number of inactive stations at *State* k. Consequently, Q_k^{AP} (the length of the AP queue) and η_k (the number of active TCP stations, including the AP) at *State* k can be calculated by

$$Q_k^{\text{AP}} \triangleq \sum_{j=0}^{W} N_k^j \cdot (W-j), \tag{1}$$

and

$$\eta_k \triangleq \begin{cases} N_{\text{STA}} - N_k^0 + 1, & N_k^W < N_{\text{STA}}, \\ N_{\text{STA}}, & N_k^W = N_{\text{STA}}. \end{cases} \tag{2}$$

Let M be the total number of states. Now, the steady-state probability of the Markov chain is given by

$$\pi_k = \lim_{v \to \infty} \text{P}\left\{\vec{N}(v) = \vec{N}_k\right\}, \text{ for each } k \in [0, M-1]. \tag{3}$$

In this Markov chain, the state transition occurs only at the end of each virtual slot, which ends with a successful transmission. A transition from a right state to a left state in Fig. 3 is caused by a station's successful transmission, while a transition from a left state to a right state is caused by a successful transmission by the AP. For example, at *State 2*, the number of active stations (including the AP) is three. Each of the two non-AP stations has exactly one TCP packet in its queue. If one of them ends the current virtual slot with a successful transmission, the transition from *State 2* to *State 1* occurs. On the other hand, if the AP ends the current virtual slot with its successful transmission, there are two possible cases. The first case is that the AP transmits a TCP packet to one of the $(N_{\text{STA}}-2)$ inactive stations. In this case, the number of active stations becomes $2+1=3$, meaning that the transition from *State 2* to *State 4* occurs. The second case is that the AP transmits a TCP packet to one of the two active stations, each of which already has one TCP packet in its queue. As a result, the number of active stations remains the same while the station that just received the TCP packet from the AP is allowed to add one more packet to its queue. This means that a transition from *State 2* to *State 5* has occurred.

C. State Transition Probabilities

Now, we derive the state transition probabilities. As discussed above, during each transition, only two elements of the state vector change their values. Each of them changes by one. This is because state transitions are triggered by a single successful transmission by either the AP or a station. Therefore, the transition probability from *State* k to *State* m is given by

$$P_{k,m} = \begin{cases} P_{\text{AP},k}^i, & \text{if } \exists \text{ a unique } i \in [0, W-1] \text{ such that} \\ & N_m^i = N_k^i - 1,\ N_m^{i+1} = N_k^{i+1} + 1, \\ P_{\text{STA},k}^i, & \text{if } \exists \text{ a unique } i \in [0, W-1] \text{ such that} \\ & N_m^i = N_k^i + 1,\ N_m^{i+1} = N_k^{i+1} - 1, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The first equation in (4) accounts for the fact that the AP ends the current virtual slot with its successful transmission. $P_{\text{AP},k}^i$ is the probability that the AP successfully transmits its TCP packet to one of the class-i stations. The second equation accounts for the fact that a station ends the current virtual slot with its successful transmission. $P_{\text{STA},k}^i$ is the probability that a class-$(i+1)$ station succeeds in transmitting its TCP packet.

The probability $P_{\text{AP},k}^i$ is given by

$$P_{\text{AP},k}^i = P_{\text{AP},k}^{\text{succ}} \cdot P_k^{\text{AP}(i)} \tag{5}$$

where

$$\begin{aligned} P_{\text{AP},k}^{\text{succ}} &= P\left(\text{AP's success} \mid \text{a tx success at } State\ k\right) \\ &= \begin{cases} 1/\eta_k, & N_k^W < N_{\text{STA}}, \\ 0, & N_k^W = N_{\text{STA}} \end{cases} \end{aligned} \tag{6}$$

and

$$\begin{aligned} P_k^{\text{AP}(i)} &= P\left(\text{AP tx to a class}-i \text{ STA} \mid \text{AP's success}\right) \\ &= \begin{cases} N_k^i (W-i)/Q_k^{\text{AP}}, & Q_k^{\text{AP}} > 0, \\ 0, & Q_k^{\text{AP}} = 0. \end{cases} \end{aligned} \tag{7}$$

Similarly, the probability $P_{\text{STA},k}^i$ is given by

$$P_{\text{STA},k}^i = P_{\text{STA},k}^{\text{succ}} \cdot P_k^{\text{STA}(i+1)} \tag{8}$$

where

$$\begin{aligned} P_{\text{STA},k}^{\text{succ}} &= P\left(\text{STA's success} \mid \text{a tx success at } State\ k\right) \\ &= 1 - P_{\text{AP},k}^{\text{succ}} \end{aligned} \tag{9}$$

and

$$\begin{aligned} P_k^{\text{STA}(i+1)} &= P\left(\text{it is a class}-(i+1) \text{ STA} \mid \text{STA's success}\right) \\ &= \begin{cases} N_k^{i+1}/(\eta_k - 1), & N_k^0 < N_{\text{STA}}, \\ 0, & N_k^0 = N_{\text{STA}}. \end{cases} \end{aligned} \tag{10}$$

D. Average Number of Active TCP Stations

Since the length of the virtual slot (i.e., the sojourn time) in each state varies, the average number of active stations (excluding the AP) is

$$E\left[\#\text{ of active STAs}\right] = \sum_{k=0}^{M-1} (N_{\text{STA}} - N_k^0) p_k \tag{11}$$

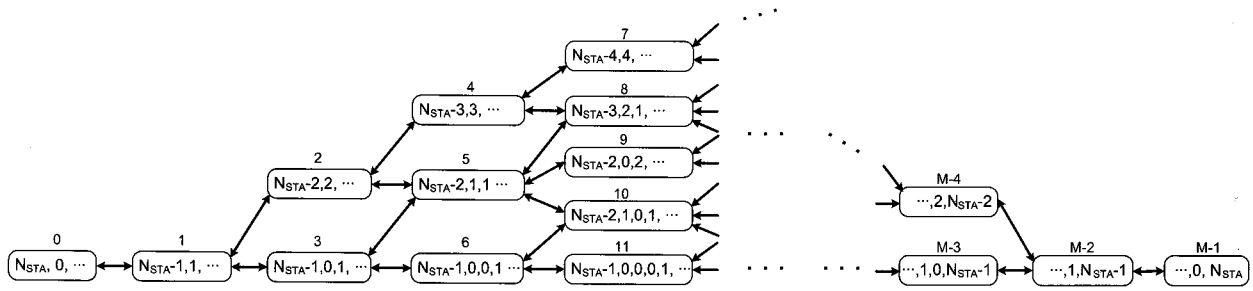


Fig. 3. Our Markov chain model to study TCP dynamics in 802.11 WLANs.

where

$$p_k = \pi_k \mu_k \Big/ \left(\sum_{j=0}^{M-1} \pi_j \mu_j \right)$$

where μ_k is the sojourn time at *State* k. In the case of download TCP flows, μ_k is given by

$$\begin{aligned} \mu_k =& E\left[N_{\mathrm{STA},k}^{\mathrm{col}}\right] T_{\mathrm{col}}^{\mathrm{ack}} + E\left[N_{\mathrm{AP},k}^{\mathrm{col}}\right] T_{\mathrm{col}}^{\mathrm{data}} \\ &+ \left(E\left[N_k^{\mathrm{col}}\right] + 1\right) \left(\mathrm{DIFS} + E\left[T_k^{\mathrm{idle}}\right]\right) \\ &+ E\left[T_k^{\mathrm{data}}\right] + \mathrm{SIFS} + T_{\mathrm{ACK}}. \end{aligned} \tag{12}$$

Here, $T_{\mathrm{col}}^{\mathrm{ack}}$ and $E\left[N_{\mathrm{STA},k}^{\mathrm{col}}\right]$ are, respectively, the collision time and the average number of collisions due to contention among uplink TCP ACK packets. $T_{\mathrm{col}}^{\mathrm{data}}$ and $E\left[N_{\mathrm{AP},k}^{\mathrm{col}}\right]$ are, respectively, the collision time and the average number of collisions due to contention between a downlink TCP data and uplink TCP ACKs. Moreover, $E\left[T_k^{\mathrm{data}}\right]$ and T_{ACK} are the average TCP packet transmission time and MAC-layer ACK frame transmission time, respectively. $E\left[N_k^{\mathrm{col}}\right]$ and $E\left[T_k^{\mathrm{idle}}\right]$ are the average number of collisions and the average idle time preceding a collision or a successful transmission, respectively. On the other hand, in the case of upload TCP, μ_k can be calculated by replacing $T_{\mathrm{col}}^{\mathrm{ack}}$ by $T_{\mathrm{col}}^{\mathrm{data}}$ in (12), because at least one of the colliding packets is a TCP data, and hence the collision time is always $T_{\mathrm{col}}^{\mathrm{data}}$.

In the following, we derive individual elements used in (12). We assume that the AP and each station access the medium with attempt probability τ whose value will be derived in Section III-F. $E\left[N_k^{\mathrm{col}}\right]$, $E\left[N_{\mathrm{AP},k}^{\mathrm{col}}\right]$, and $E\left[N_{\mathrm{STA},k}^{\mathrm{col}}\right]$ can be expressed by

$$\begin{aligned} E\left[N_k^{\mathrm{col}}\right] &= \sum_{i=0}^{\infty} i \cdot P\left(N_k^{\mathrm{col}} = i\right), \\ E\left[N_{\mathrm{AP},k}^{\mathrm{col}}\right] &= \sum_{j=0}^{\infty} j \cdot P\left(N_{\mathrm{AP},k}^{\mathrm{col}} = j\right), \\ E\left[N_{\mathrm{STA},k}^{\mathrm{col}}\right] &= E\left[N_k^{\mathrm{col}}\right] - E\left[N_{\mathrm{AP},k}^{\mathrm{col}}\right] \end{aligned} \tag{13}$$

where $P\left(N_k^{\mathrm{col}} = i\right)$ and $P\left(N_{\mathrm{AP},k}^{\mathrm{col}} = j\right)$ are the probability that the total number of collisions is i and the probability that the number of collisions due to the AP's transmissions is j, respectively, which are given by

$$\begin{aligned} P\left(N_k^{\mathrm{col}} = i\right) &= \left(P_k^{\mathrm{col}}\right)^i P_k^{\mathrm{succ}}, \\ P\left(N_{\mathrm{AP},k}^{\mathrm{col}} = j\right) &= \left(P_{\mathrm{AP},k}^{\mathrm{col}}\right)^j P_k^{\mathrm{succ}}. \end{aligned} \tag{14}$$

P_k^{succ} is the conditional successful transmission probability at *State* k given that at least one station (including the AP) transmits, and can be calculated by

$$P_k^{\mathrm{succ}} = \mathrm{P}\left(N_k^{\mathrm{tx}} = 1 \,|\, N_k^{\mathrm{tx}} \geq 1\right) = \frac{\eta_k \cdot \tau \left(1-\tau\right)^{\eta_k - 1}}{1 - \left(1-\tau\right)^{\eta_k}}. \tag{15}$$

Here, N_k^{tx} denotes the number of transmitting stations at *State* k. P_k^{col} is the conditional collision probability given that at least one station transmits, which is $P_k^{\mathrm{col}} = 1 - P_k^{\mathrm{succ}}$. Moreover, $P_{\mathrm{AP},k}^{\mathrm{col}}$ is the conditional probability that AP's transmission fails given that at least one station (including the AP) transmits, which is

$$P_{\mathrm{AP},k}^{\mathrm{col}} = \begin{cases} \frac{\tau\left(1-(1-\tau)^{\eta_k - 1}\right)}{1-(1-\tau)^{\eta_k}}, & N_k^W < N_{\mathrm{STA}}, \\ 0, & N_k^W = N_{\mathrm{STA}}. \end{cases} \tag{16}$$

The average idle time preceding a collision or a successful transmission in *State* k can be calculated as

$$\begin{aligned} E[T_k^{\mathrm{idle}}] &= \mathrm{aSlotTime} \cdot \sum_{i=0}^{\infty} i \cdot P\left(N_k^{\mathrm{tx}} > 0\right) \cdot \left(P\left(N_k^{\mathrm{tx}} = 0\right)\right)^i \\ &= \mathrm{aSlotTime} \cdot \frac{\left(1-\tau\right)^{\eta_k}}{1-\left(1-\tau\right)^{\eta_k}}. \end{aligned} \tag{17}$$

In the case of download TCP, the average TCP packet transmission time can be expressed as

$$E\left[T_k^{\mathrm{data}}\right] = T_{\mathrm{TCP\,DATA}} \cdot P_{\mathrm{AP},k}^{\mathrm{succ}} + T_{\mathrm{TCP\,ACK}} \cdot P_{\mathrm{STA},k}^{\mathrm{succ}} \tag{18}$$

where $T_{\mathrm{TCP\,DATA}}$ and $T_{\mathrm{TCP\,ACK}}$ are TCP data and TCP ACK transmission durations, respectively. In the case of upload TCP, the positions of $P_{\mathrm{AP},k}^{\mathrm{succ}}$ and $P_{\mathrm{STA},k}^{\mathrm{succ}}$ are exchanged in (18).

E. Aggregate TCP Throughput

In the p-persistent CSMA/CA model [9], the aggregate throughput (in bits/second) is given by

$$\rho = \frac{\text{average length of a data frame (bits)}}{\text{average duration of a virtual slot (sec)}}. \tag{19}$$

In our analysis, under the assumption that there exist a fixed number of active stations in a given state, the download TCP throughput in *State* k can be derived as

$$\rho_k = \frac{L_{\mathrm{TCP\,DATA}} \cdot P_{\mathrm{AP},k}^{\mathrm{succ}}}{\mu_k} \tag{20}$$

Table 2. Algorithm to estimate τ from CWmin at each state.

At *State k*

Step 0: Initialization

for $(i = 0, ..., n_r)$

$cw_i = \min\left(\text{CWmax}, 2^i(\text{CWmin}+1)-1\right)$

$\overline{CW} = \text{CWmin}$

Step 1: Calculate τ and save the current $\overline{CW}$

$\tau = \frac{2}{\overline{CW}+1}$, $\overline{CW}_{\text{old}} = \overline{CW}$

Step 2: Calculate $\overline{CW}$

for $(i = 0, ..., n_r)$

$P(CW = cw_i) = p_0(1-p_{\text{col},k})(p_{\text{col},k})^i$

$\overline{CW} = \sum_{i=0}^{n_r} cw_i P(CW = cw_i)$

Step 3: Determine whether the iteration ends

If $|\overline{CW} - \overline{CW}_{\text{old}}| < \varepsilon$, then $\{\tau = \frac{2}{\overline{CW}+1}$, **return** $\tau\}$

Otherwise, **go to Step 1**

where $L_{\text{TCP DATA}}$ is the TCP data (i.e., segment) size (in bits). If we consider the upload TCP throughput, we use $P^{\text{succ}}_{\text{STA},k}$ instead of $P^{\text{succ}}_{\text{AP},k}$ in the above equation. Finally, the aggregate TCP throughput in an infrastructure-based WLAN can be determined as follows:

$$\rho_{\text{TCP}} = \sum_{i=0}^{M-1} \rho_i \cdot p_i. \tag{21}$$

F. Estimation of τ from CWmin in Each State

We now extend the average contention window estimation algorithm in [9] to obtain the slot access probability τ. We assume that $\tau = 1/(\overline{B}+1)$ where $\overline{B}$ is the average number of backoff slots, and is equal to $\left(\overline{CW}+1\right)/2$ [9]. Because the average contention window size $\overline{CW}$ is determined by CWmin and the number of active stations in a given state, we can estimate τ from CWmin in each state using the iterative algorithm shown in Table 2.

As shown in Table 2, n_r is the frame retry limit at both the AP and stations. At *State k*, the algorithm determines the average contention window size iteratively. $p_{\text{col},k}$ in Step 2 is the probability that a transmission of the AP or a station collides, and is given by

$$p_{\text{col},k} = 1 - (1-\tau)^{\eta_k - 1}. \tag{22}$$

p_0 is the normalization factor and can be obtained based on the fact that $\sum_{i=0}^{n_r} P(CW = cw_i) = 1$. At the end of each iteration, if the difference between the current average contention window size ($\overline{CW}$) and its previous value ($\overline{CW}_{\text{old}}$) is less than ε, the algorithm terminates and returns the τ value.

IV. ANALYTICAL AND SIMULATION STUDIES

In this section, we validate our model via ns-2 simulation [12]. We consider IEEE 802.11b/a WLANs with a single AP and multiple stations which either download or upload large data files to remote FTP servers. Fig. 4 shows the network topology. The round trip time (RTT) of TCP packets in the wireline network is referred to as the *wireline delay*. TCP NewReno is used in our simulation. Table 3 summarizes the parameters used to produce analytical and simulation results.

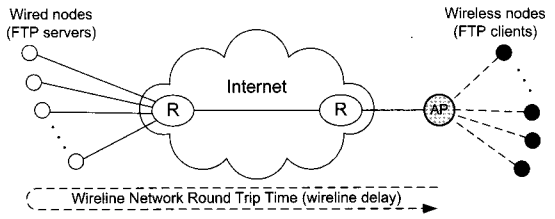


Fig. 4. Network topology in the analytical and simulation studies.

Table 3. Parameters used to produce analytical and simulation results.

Parameters	802.11b	802.11a
Slot time	20 μs	9 μs
SIFS, DIFS	10 μs, 50 μs	16 μs, 34 μs
CWmin, CWmax	31, 1023	15, 1023
Data, ACK rates	11 Mb/s, 2 Mb/s	54 Mb/s, 24 Mb/s
PHY overhead	192 μs	20 μs
MAC ACK length	14 bytes	
MAC overhead	28 bytes	
TCP data frame	Data (1460 bytes) + TCP/IP headers (40 bytes) + SNAP* header (8 bytes) + MAC/PHY overheads	
TCP ACK frame	TCP/IP headers (40 bytes) + SNAP header (8 bytes) + MAC/PHY overheads	

* When an IP datagram is transferred over 802.11 WLANs, it is typically encapsulated in an IEEE 802.2 sub-network access protocol (SNAP) packet.

We first investigate how the average number of active TCP stations and the aggregate TCP throughput are affected by the maximum TCP receive window size and the total number of TCP stations in an infrastructure-based 802.11 WLAN. We assume immediate TCP ACK, large queue size, no TCP ACK processing delay, no packet loss, and no wireline delay. Then we show that the default minimum contention window (CWmin) size is too large to maximize the aggregate TCP throughput. Finally, we study the effects of the above parameters (i.e., delayed TCP ACK, queue size, TCP ACK processing delay, packet loss,and wireline delay) on the aggregate TCP throughput in Sections IV-D, IV-E, IV-F, and IV-G.

A. Average Number of Active TCP Stations

Figs. 5–7 show the average number of active TCP stations (excluding the AP) from our analysis and ns-2 simulations with 802.11b/a PHYs. We first observe that the average number of active TCP stations remains very small (i.e., around one) in all simulated scenarios while analytical and simulation results match very well. When there are at least three stations in the network, the average number of active TCP stations remains almost constant for both upload and download cases, regardless of the maximum TCP receive window size.[2] This is due to (i) the closed-loop nature of TCP flow control and (ii) the bottleneck downlink (i.e., AP-to-station) transmissions in the

[2]Actually, typical TCP configurations in the MS Windows XP operating system use a 12-packet (or 17520-byte) receive window.

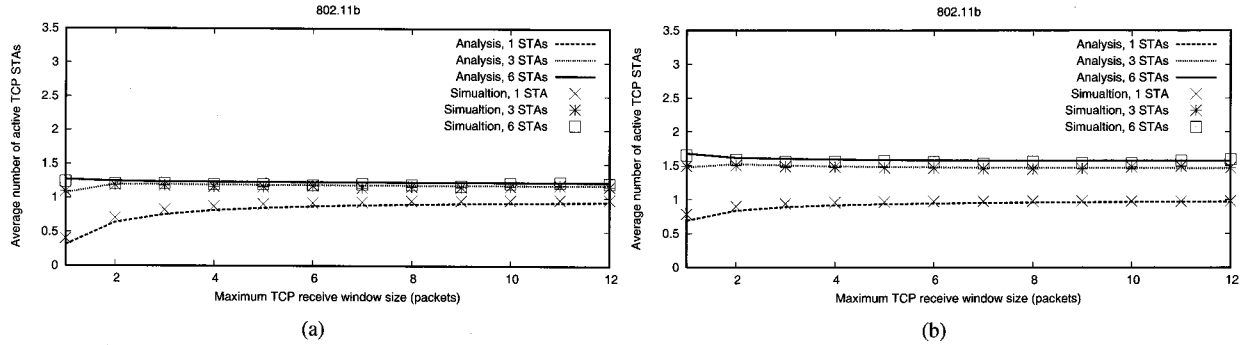


Fig. 5. Average number of active TCP stations vs. maximum TCP receive window size (W) in an 802.11b WLAN: (a) Download TCP and (b) uploadTCP.

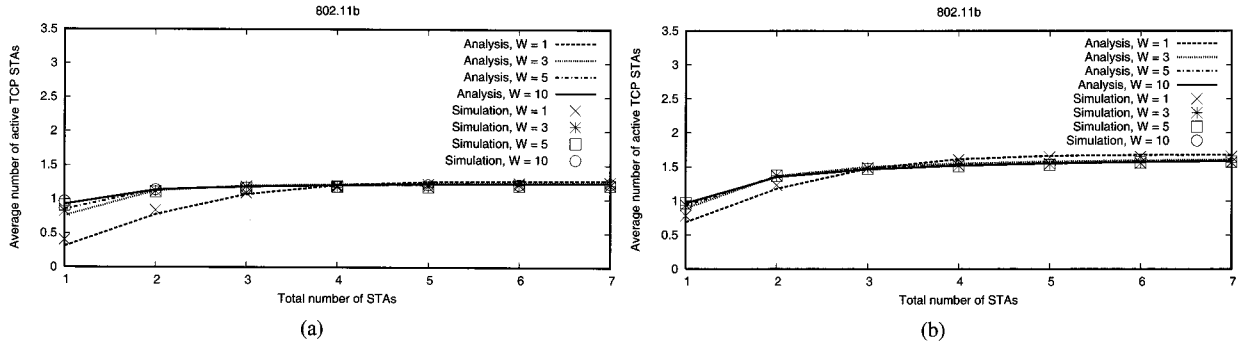


Fig. 6. Average number of active TCP stations vs. total number of TCP stations in an 802.11b WLAN: (a) Download TCP and (b) upload TCP.

infrastructure-based WLANs, as we discussed in Section I. In an 802.11 DCF-based WLAN, the AP is the bottleneck because it has the same channel access priority as stations but needs to serve multiple TCP flows. These results are almost the same as those presented in [4], where the number of active stations (including the AP) is evaluated via simulations and experiments but without theoretical analysis.

On the other hand, when the maximum TCP receive window size is one (i.e., $W = 1$) and when there is only one TCP station in the network (i.e., $N_{\text{STA}} = 1$), the average number of active TCP stations is much smaller than one. In this case, the station and the AP do not contend with each other because the number of outstanding TCP packets is just one; hence either the AP or the station can access the channel but not at the same time. As a result, the idle backoff time occupies a considerable portion of the channel time; hence the number of active TCP stations becomes much smaller than one.

In Figs. 5 and 6, we observe that the average number of active TCP stations in the case of upload is slightly larger than that in the download case. It is due to the fact that the transmission time of a station (i.e., TCP data transmission time) is larger than that in the download case (i.e., TCP ACK transmission time); hence the time duration when stations are contending for the channel is larger than that in the download case due to longer collision and transmission times.

B. Aggregate TCP Throughput

Fig. 8 shows the aggregate download TCP throughput with 802.11b/a PHYs. We observe that the numerical results are almost the same as the simulation results in both cases. The aggregate TCP throughput is almost independent of the total number of TCP stations (i.e., N_{STA}) and the maximum TCP receive window size in an infrastructure-based WLAN. This is due to the fact that the average number of active TCP stations remains nearly constant irrespective of the total number of TCP stations and the maximum TCP receive window size, as shown in Figs. 5–7.

When the number of stations is one (i.e., $N_{\text{STA}} = 1$), the throughput is slightly low. This is consistent with our earlier observations in Figs. 6(a) and 7(b) that the number of active stations is much smaller than one when $N_{\text{STA}} = 1$. That is, the lower throughput means that the portion of idle backoff time in the channel time is larger than in other cases (i.e., when $N_{\text{STA}} > 1$) due to smaller number of active stations.

C. Optimal CWmin for TCP Throughput Maximization

From Section IV-B, we know that the aggregate TCP throughput is almost independent of the total number of TCP stations (i.e., N_{STA}). In this section, we study the effects of CWmin on the aggregate TCP throughput. In Fig. 9, we plot the relation between the aggregate TCP throughput and CWmin from both numerical analysis and simulation results. We fix CWmax to 1023 in all cases. Here, we assume that there exist only download TCP stations in the network; optimal CWmin values for the upload case can be determined in a similar way.

As shown in Fig. 9, the optimal CWmin values for 802.11b and 802.11a are about 17 and 11 in the case of 1460-byte TCP data, respectively, while the default values (i.e., CWmin = 31 for 802.11b and CWmin = 15 for 802.11a) are too large to maximize the aggregate TCP throughput. Moreover, when the size of

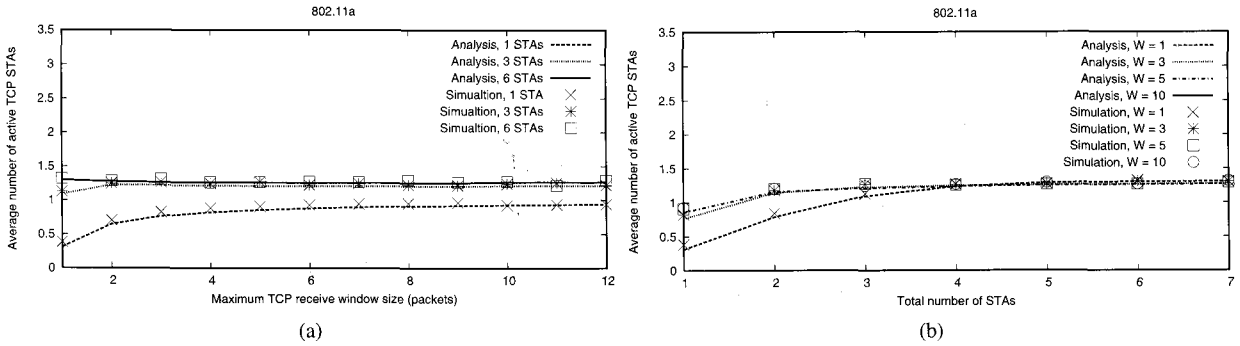


Fig. 7. Average number of active TCP STAs vs. (a) Maximum TCP receive window size and (b) total number of TCP stations when only download TCP flows exist in an 802.11a WLAN.

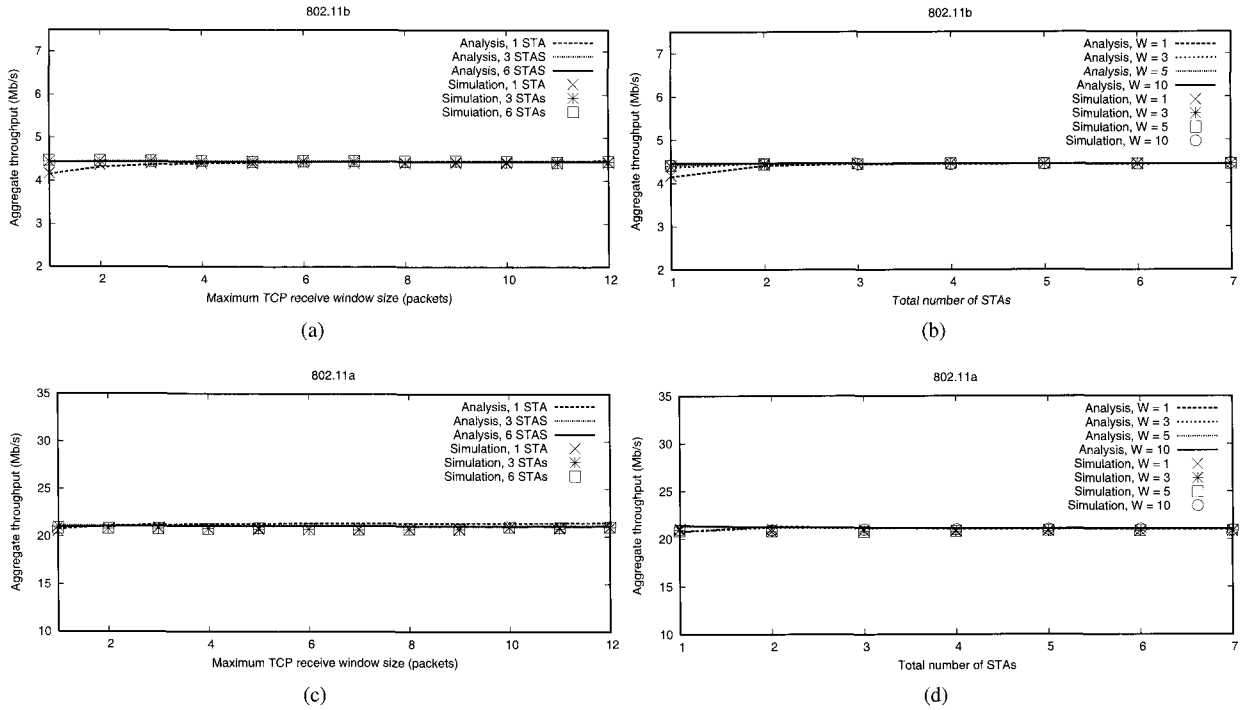


Fig. 8. Aggregate throughput when only download TCP flows exist in a WLAN: (a) 802.11b – the effect of maximum TCP receive window size, (b) 802.11b – the effect of total number of TCP STAs, (c) 802.11a – the effect of maximum TCP receive window size, and (d) 802.11a – the effect of total number of TCP STAs.

TCP data is small (i.e., 460 bytes), we observe that even smaller CWmin (i.e., 11 and 6 for 802.11b and 802.11a, respectively) are desired. This is due to the fact that the collision overhead decreases as the size of TCP data decreases. Unfortunately, adapting the CWmin value is not allowed in 802.11. However, the emerging 802.11e allows to control channel access parameters, including CWmin [13]. In fact, our work can be extended to study the effects of various channel access parameters on the aggregate TCP throughput in the emerging 802.11e WLAN.

D. *Effects of Delayed ACK*

In this section, we investigate the effect of delayed TCP ACK on the aggregate TCP throughput via simulations. Note that we have assumed for the analysis that immediate TCP ACK is used instead of delayed TCP ACK as mentioned in Section III-A. As shown in Fig. 10, obviously, the throughput performance is improved via delayed TCP ACK due to the reduced ACK transmission overhead. However, we observe that the throughput of delayed TCP ACK is also independent of the number of TCP stations. This is due to the facts that (i) the TCP senders can send TCP data only when it receives TCP ACK regardless of the ACK scheme and (ii) the AP is still the bottleneck. We might be able to approximate the performance of the delayed TCP ACK by simply scaling our analytical results with the immediate TCP ACK considering the TCP ACK overhead.

E. *Effects of Queue Size*

In this section, we study the effect of queue sizes of both the AP and stations on the aggregate TCP throughput via simulations. As shown in Fig. 11, the aggregate TCP throughput remains flat independent of the queue sizes when the queue size is larger than 10. Note that the queue sizes of most off-the-shelf WLAN devices are much larger than 10 (in packets) [14]. This means that assumption [A1] in Section III-A hardly affects the

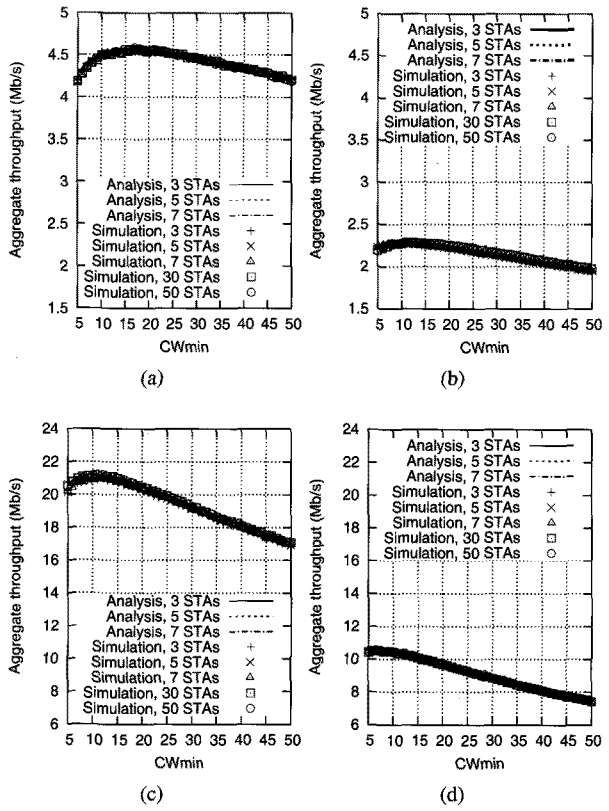


Fig. 9. The effect of CWmin on the aggregate TCP throughput: (a) 802.11b, TCP data size = 1460 bytes, (b) 802.11b, TCP data size = 460 bytes, (c) 802.11a, TCP data size = 1460 bytes, and (d) 802.11a, TCP data size = 460 bytes.

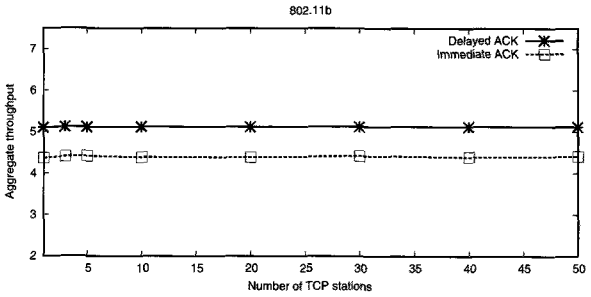


Fig. 10. The effects of delayed ACK when all stations download data from FTP servers.

aggregate TCP throughput performance.

So far, we have evaluated the effects of CWmin values on the aggregate TCP throughput without considering TCP ACK processing delay, wireline delay, and packet loss. Next, we study the effects of these parameters via simulations.

F. Effects of TCP ACK Processing Delay

Fig. 12 shows the effects of TCP ACK processing time on the aggregate TCP throughput. We assume no wireline delay in this scenario. In general, we observe that, with non-zero TCP ACK processing delay, the aggregate throughput deviates little from our analysis result (i.e., 4.46 Mb/s) by at most 5%. This means that our analysis is reasonably accurate even with the simplifying assumption of no TCP ACK processing delay. As shown

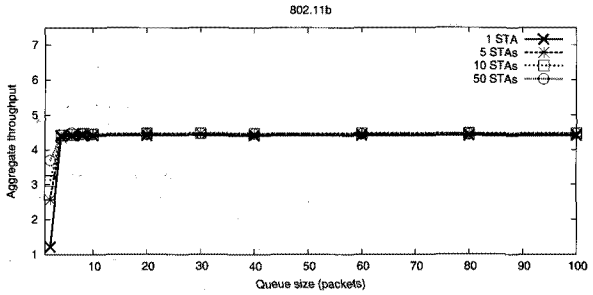


Fig. 11. The effects of queue sizes of both the AP and stations when all stations download data from FTP servers.

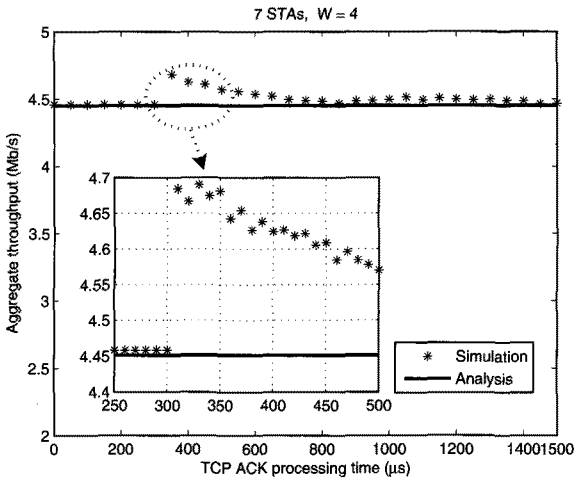


Fig. 12. The effects of TCP ACK processing delay (with no wireline delay) when all stations download data from FTP servers.

in the figure, the throughput increases when the TCP ACK processing delay is around 308 μs. This is due to the *immediate access* behavior that was discussed in Section II-A. The reason can be explained as follows. Fig. 13 shows the timing diagram of the AP and a station that receives a TCP data from the AP. If the TCP ACK is generated within duration **A** in Fig. 13, the MAC starts backoff at the end of duration **A** because it senses the channel busy. On the other hand, if the TCP ACK is generated within duration **B**, the MAC might send the TCP ACK immediately without backoff. Note that the length of duration **A** is 308 μs = SIFS (10μs) + ACK (248 μs) + DIFS (50 μs). In the latter case, in order to have immediate access, the station should satisfy the following two conditions when it receives TCP Data from the AP: (i) It has already finished the post backoff and (ii) it is not contending for the channel to transmit a packet (i.e., TCP ACK). Most stations satisfy the above conditions because they remain idle for most of the time while waiting for TCP data from the AP due to AP bottleneck and TCP flow control.

G. Effects of Wireline Delay and Packet Loss

Fig. 14 shows the effects of wireline delay (from 0 to 200 ms) and the end-to-end packet loss rate (from 0.1% to 5% according to [15]) on the aggregate TCP throughput. Results with the maximum TCP receive window sizes of $W = 4$ and $W = 12$ are shown in Figs. 14(a) and 14(b), respectively. It is clear from the figure that our model produces accurate results when the wireline delay is small and the packet loss rate is low, i.e., when the

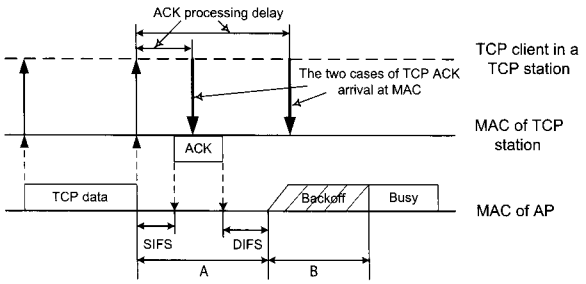


Fig. 13. Illustration of the TCP ACK processing delay in a station when it downloads data from a server via the AP.

WLAN is the bottleneck in the network.

However, when the wireline delay gets larger, the WLAN is not the bottleneck anymore. That is, the aggregate TCP throughput is dominantly determined by the wireline delay. We observe that the aggregate TCP throughput decreases as the wireline delay increases. Moreover, packet loss accelerates the throughput degradation. We observe that the higher the packet loss is, the more quickly the aggregate throughput decreases as the wireline delay increases. The reason is as follows. The high packet loss rate causes TCP timeout and in turn decreases the number of outstanding TCP packets, because the TCP send window shrinks to one MSS after timeout.

Fig. 14(b) shows that a larger maximum TCP receive window results in a bigger applicable region (i.e., the flat region) for our model. This is because, with a larger maximum TCP receive window, more outstanding TCP packets are allowed in the network, hence WLAN may remain as the bottleneck even in the presence of larger wireline delay and/or higher packet loss rate.

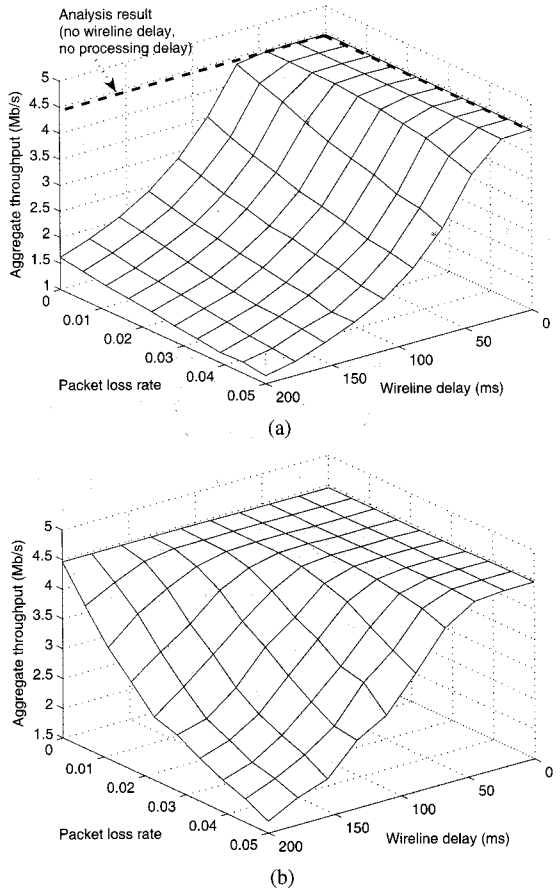


Fig. 14. The effects of wireline delay and packet loss on the aggregate TCP throughput when all stations download data from FTP servers. W is the maximum TCP receive window size (in packets): (a) TSTAs, 802.11b PHY, $W = 4$ and (b) TSTAs, 802.11b PHY, $W = 12$.

V. CONCLUSION

In this paper, we conduct a comprehensive analysis of the TCP dynamics over infrastructure-based IEEE 802.11 WLANs. We study the effects of the total number of TCP stations and the maximum TCP receive window size on the average number of the active TCP stations and the aggregate TCP throughput. Moreover, we show that the default minimum contention window sizes are too large to maximize the TCP throughput in both 802.11b and 802.11a WLANs. Finally, the assumptions used in our analytical model are evaluated via simulation, and the results show that our model is reasonably accurate when the wireline delay is small and/or the packet loss rate is low.

REFERENCES

[1] IEEE, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, IEEE Std 802.11-1999, 1999.

[2] H. Jiang and C. Dovrolis, "Why is the Internet traffic bursty in short time scales?," in *Proc. ACM SIGMETRICS*, June 2005.

[3] J. Yu, S. Choi, and J. Lee, "Enhancement of VoIP over IEEE 802.11 WLAN via dual queue strategy," in *Proc. IEEE ICC*, June 2004.

[4] S. Choi, K. Park, and C. Kim, "Performance impact of interlayer dependence in infrastructure WLANs," *IEEE Trans. Mobile Computing*, vol. 5, no. 7, July 2006.

[5] A. A. Kherani and R. Shorey, "Modelling TCP performance in multihop 802.11 networks with randomly varying channel," in *Proc. WILLOPAN*, Jan. 2006.

[6] C. Burmeister and U. Killat, "TCP over rate-adaptive WLAN–An analytical model and its simulative verification," in *Proc. IEEE WoWMoM*, June 2006.

[7] J. Choi, K. Park, and C. Kim, "Cross-layer analysis of rate adaptation, dcf and tcp in multi-rate WLANs," in *Proc. IEEE INFOCOM*, Apr. 2007.

[8] R. Bruno, M. Conti, and E. Gregori, "Analytical modeling of TCP clients in Wi-Fi hot spot networks," in *Proc. IFIP Networking*, May 2004.

[9] F. Cali, M. Conti, and E. Gregori, "Dynamic tuning of the IEEE 802.11 protocol," *IEEE/ACM Trans. Networking*, vol. 8, no. 6, Dec. 2000.

[10] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, MA: Addison-wesley, 1994, vol. 1.

[11] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," *Computer Commun. Rev.*, vol. 26, no. 3, pp. 5–21, July 1996.

[12] The Network Simulator – ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[13] IEEE, Supplement to Part 11: Medium Access Control (MAC) Enhancements for Quality of Service (QoS), IEEE Std 802.11e, Nov. 2005.

[14] F. Li, M. Li, R. Lu, H. Wu, M. Claypool, and R. Kinicki, "Measuring queue capacities of IEEE 802.11 wireless access points," in *Proc. IEEE BROADNETS*, Sept. 2007.

[15] Y. Yamasaki, H. Shimonishi, and T. Murase, "Statistical estimation of TCP packet loss rate from sampled ACK packets," in *Proc. IEEE GLOBECOM*, Dec. 2005.

[16] IEEE, Supplement to Part 11: Higher-speed Physical Layer Extension in the 2.4 GHz Band, IEEE Std. 802.11b-1999, 1999.

[17] IEEE, Supplement to Part 11: Higher-speed Physical Layer Extension in the 5 GHz Band, IEEE Std. 802.11a-1999, 1999.

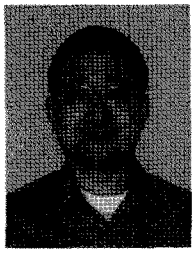
[18] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, Mar. 2000.

[19] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11," in *Proc. IEEE INFOCOM*, Mar. 2001.

[20] I. Tinnirello and S. Choi, "Temporal fairness provisioning in multi-rate contention-based 802.11e WLANs," in *Proc. IEEE WoWMoM*, June 2005.

Jeonggyun Yu received his B.E. degree in the School of Electronic Engineering from Korea University in August 2002 and Ph.D. degree at the School of Electrical Engineering, Seoul National University (SNU) in February 2009, respectively. He is currently a senior engineer with Samsung Electronics Co., LTD, developing mobile/wireless access system. His research interests include QoS support, algorithm development, performance evaluation for wireless networks.

Sunghyun Choi is currently an Associate Professor at the School of Electrical Engineering, Seoul National University (SNU), Seoul, Korea. Before joining SNU in September 2002, he was with Philips Research USA, Briarcliff Manor, New York, USA as a *Senior Member Research Staff and a project leader* for three years. He received his B.S. (summa cum laude) and M.S. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST) in 1992 and 1994, respectively, and received Ph.D. at the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor in September, 1999. His current research interests are in the area of wireless/mobile networks with emphasis on wireless LAN/MAN/PAN, next-generation mobile networks, mesh networks, cognitive radios, resource management, data link layer protocols, and cross-layer approaches. He authored/coauthored over 120 technical papers and book chapters in the areas of wireless/mobile networks and communications. He has co-authored (with B. G. Lee) a book "Broadband Wireless Access and Local Networks: Mobile WiMAX and WiFi," Artech House, 2008. He holds 15 US patents, nine European patents, and nine Korea patents, and has tens of patents pending. He has served as a General Co-Chair of COMSWARE 2008, and a Technical Program Committee Co-Chair of ACM Multimedia 2007, IEEE WoWMoM 2007 and IEEE/Create-Net COMSWARE 2007. He was a Co-Chair of Cross-Layer Designs and Protocols Symposium in IWCMC 2006, 2007, and 2008, the workshop co-chair of WILLOPAN 2006, the General Chair of ACM WMASH 2005, and a Technical Program Co-Chair for ACM WMASH 2004. He has also served on program and organization committees of numerous leading wireless and networking conferences including IEEE INFOCOM, IEEE SECON, IEEE MASS, and IEEE WoWMoM. He is also serving on the editorial boards of IEEE Transactions on Mobile Computing, ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), and Journal of Communications and Networks (JCN). He is serving and has served as a guest editor for IEEE Journal on Selected Areas in Communications (JSAC), IEEE Wireless Communications, Pervasive and Mobile Computing (PMC), ACM Wireless Networks (WINET), Wireless Personal Communications (WPC), and Wireless Communications and Mobile Computing (WCMC). He gave a tutorial on IEEE 802.11 in ACM MobiCom 2004 and IEEE ICC 2005. From 2000 to 2007, he was a Voting Member of IEEE 802.11 WLAN Working Group. He has received a number of awards including the Young Scientist Award awarded by the President of Korea (2008); IEEK/IEEE Joint Award for Young IT Engineer (2007); the Outstanding Research Award (2008) and the Best Teaching Award (2006) both from the College of Engineering, Seoul National University; the Best Paper Award from IEEE WoWMoM 2008; and Recognition of Service Award (2005, 2007) from ACM. He was a recipient of the Korea Foundation for Advanced Studies (KFAS) Scholarship and the Korean Government Overseas Scholarship during 1997–1999 and 1994–1997, respectively. He is a Senior Member of IEEE, and a Member of ACM, KICS, IEEK, and KIISE.



Daji Qiao is currently an Assistant Professor in the Department of Electrical and Computer Engineering, Iowa State University, Ames, Iowa. He received his Ph.D. degree in Electrical Engineering-Systems from The University of Michigan, Ann Arbor, Michigan, in February 2004. His current research interests include modeling, analysis and protocol/algorithm design for various types of wireless/mobile networks, including IEEE 802.11 Wireless LANs, mesh networks, and sensor networks. He is a Member of IEEE and ACM.