Performance Tuning of Infrastructure-Mode Wireless LANs

Yigal Bejerano, Hyoung-Gyu Choi, Seung-Jae Han, Thyaga Nandagopal

Performance Tuning of Infrastructure-Mode Wireless LANs

Yigal Bejerano† **Hyoung-Gyu Choi**‡ **Seung-Jae Han**‡ **Thyaga Nandagopal**†
† Bell-Labs, Alcatel-Lucent, NJ, USA. ‡ Yonsei University, Seoul, Korea.

***Abstract*—Conventional wisdom about 802.11 WLANs dictates that as the number of active users increases, the contention windows (CW) of all the contending users needs to increase to prevent collision losses. However, given the increasing data rates of 802.11 standards, the fraction of time spent not transmitting information (as a result of back-off) is also increasing, leading to reduced throughput when compared to the theoretical maximum. While there appears to be an inherent conflict between reducing collisions by increasing the CW and reducing overhead by decreasing the CW, we demonstrate in this paper that this conflict can be eliminated in Infrastructure-Mode wireless LANs (I-WLAN) where downlink TCP and UDP flows dominate. Unlike the traditional trend of WLAN performance analysis that assumes that all the stations are greedy, we consider the case in which only the access point (AP) is greedy while the mobile users in the cell react to the information sent to them. This setting captures the traffic patterns in I-WLANs more accurately as users typically utilize their WLAN connections predominantly for downloading information through TCP (i.e., web browsing) and UDP (i.e., video streaming) connections. We show via analysis and extensive experimentation that by changing the minimum CW at the APs and the users, we can achieve 25 - 30% greater throughput than the typically recommended 802.11 settings. We achieve this without requiring any changes to the MAC or PHY, and utilizing only the standard features present on current-generation 802.11 chip-sets.**

Keywords: IEEE 802.11 WLANs, Performance Analysis.

I. Introduction

Last hop wireless data networks are dominated by the IEEE 802.11 family [1] of standards, that support higher and higher data rates. Recent proposals for the 802.11 standards aim for 600 Mbps peak data transfer rates on the PHY layer[2]. On the surface, improvements in the PHY data rates appear to translate directly to the end-to-end throughput experienced by the user. However, due to the various overheads required by the distributed nature of the 802.11 Medium Access Control protocol, the actual throughput experienced at the transport layer (TCP/UDP) is nearly half of the physical layer rate. For example, when we use the default recommended settings for 802.11a/g, the throughput at the TCP layer is only 23 Mbps (see Section III for details) for a single pair of nodes.

Many papers have focused on analyzing the performance of 802.11 wireless networks [3], [4], [5], [6], [7], with the most notable being the paper by Cali et.al. [3]. These papers commonly analyze the *saturated nodes* case, where all the nodes are backlogged and contend for the channel all the time. A few other recent papers have analyzed the *unsaturated node* case where *every node* is not greedy all the time[8]. In reality, the most common use-case of IEEE 802.11 WLANs is the hot-spot scenario, which does not fall in either of the above categories considered in contemporary research. In order to see this, consider a typical hot-spot with an Access Point (AP) and a bunch of users. Users access web content predominantly, with very few applications sending upstream data (e.g., file sharing, audio/video conferencing). The traffic is primarily a mix of TCP flows and streaming UDP flows on the downlink, with some occasional TCP or low bit-rate UDP flows on the uplink, as seen at the AP. We identify these hot-spot settings as *Infrastructure-mode WLANs* (I-WLAN).

In I-WLANs, the AP is the node that is backlogged more than any other user, and contends for the channel nearly all the time. Users on the other hand have very few upstream flows to send to the AP. Even when the upstream flows are all TCP flows, the AP has to contend for the channel in order to send the TCP acknowledgements (ACKs) to the users. This implies that every user essentially competes with the AP. We consider these I-WLANs in detail and describe how to improve the overall performance on these networks, supported by an extensive evaluation via analysis and experimentation on a 802.11a/g testbed.

Our key contributions are as follows.

1) We show that TCP flows with their inherent ack-based self-clocking regulate the number of contending users in the channel, regardless of the number of TCP flows served on the downlink.
2) We show that for the case when downlink flows dominate the channel, the AP can set its minimum congestion window (CW_{min}) to a really small value that is smaller than the current recommendations of the 802.11 standards, regardless of the number of TCP/UDP flows on the downlink, or the number of users in the system receiving these downlink flows.
3) We identify a very interesting phenomenon wherein if the users send TCP ACKs faster by using a smaller CW_{min} than that of the AP, the throughput of the TCP session increases noticeably.
4) We present the ideal CW_{min} settings for handling a mix of upstream and downstream TCP and UDP flows when the AP is always backlogged and only a few users are actively sending upstream traffic at any time. We show that this joint transport-MAC layer optimized setting does much better than a pure MAC layer approach.
5) We present throughput analysis that allows us to determine the optimal value of the CW_{min} at the AP for any mix of downlink/uplink TCP/UDP flows, and confirm

via experimentation on a 802.11 testbed that our analysis matches with the experimental results.

Our results counter the conventional notion that the more users in a hot-spot, the higher the average back-off window should be[3], [9]. Thus, we are able to achieve performance gains of up to 25 - 30% greater throughput over recommended 802.11a/g settings in a I-WLAN for the case of TCP and even more for UDP traffic on the downlink. Our proposed changes do not require any changes in PHY or MAC layers of IEEE 802.11 standards, and utilizes existing features of WLAN chipsets[10]. The work in [11] is most related to our contributions here. They analyze the TCP throughput performance in a hot-spot network, and show that the number of contending users is nearly a constant in a I-WLAN. In this work, we analyze both TCP and UDP downlink flows, and present methods to exploit the reduced contention levels in I-WLANs. Our analysis method is simpler and more flexible than the one in [11], allowing us to model more complex scenarios. Our results are also backed up by experimentation on a real testbed, unlike the work in [11].

IEEE 802.11e[12] provides for various flow classes with variable AIFS and CW values using EDCA. The general understanding is that high priority flows should use lower AIFS and CW values than lower priority flows. However, the classification of flows is largely left to the application which could use either TCP or UDP. In this paper, we take an alternate approach and show how even with a set of generic TCP flows, we can classify uplink-ACK flows as a higher priority flow leading to improved throughput for downlink flows. This could be used as a hint for 802.11e mechanisms.

Along these lines, Leith et. al. [13] analyze the Bianchi model for competing TCP downlink + uplink flows in an AP, and suggest, based on simulations that $CW_{min} = 32$ be used everywhere, except that AIFS of 0 be used for ACKs, and the AP send ACKs out with a $CW_{min} = 2$. However, the analysis does not justify these AIFS values. They do not consider UDP traffic and do not do any experiments. We show both analytically and via actual experimentation, that, contrary to their results, we need not worry about AIFS values, but simply use different CW_{min} values for ACK traffic to achieve optimum TCP/UDP throughput, even with a mixture of TCP and UDP flows.

Many optimizations have been proposed as part of the IEEE 802.11n standard[2] to minimize the idle-time overhead. These include Frame Aggregation, Multiple Block Ack and Reduced Inter-Frame Spacing (RIFS). Combined together, they can reduce the idle time. However, they will not help much to reduce TCP ACK overhead, since it is a single cumulative ACK over multiple TCP segments and is only 72 bytes in size. This will prevent the usage of any of these features to improve TCP performance. However, by using a small CW_{min} for TCP ACKs as we show here, we can have more significant improvements in TCP throughput in 802.11n I-WLANs.

The rest of the paper is organized as follows. In Section II, we provide an overview of the relevant parts of the 802.11 protocol, and describe why the transport layer performance is very poor in Section III. In Section IV, we provide a modeling approach to quantify the performance of I-WLANs. We verify our results with data from a I-WLAN testbed in Section V and conclude in Section VI.

II. Preliminary

A. IEEE 802.11 DCF Mechanism

In this paper, we consider IEEE 802.11 Wireless LAN (WLAN) operating in infrastructure mode with one access point (AP). All users use the standard DCF MAC. DCF is a distributed medium access control method that employs the exponential back-off-based Carrier Sense Multiple Access/ Collision Avoidance (CSMA/CA) mechanism. Each user (either an AP or an user) senses the wireless channel before transmission by using both physical and virtual carrier sensing mechanisms. It starts transmission only if the channel is idle for a back-off duration, measured in time slots (or simply *slots*), which is randomly chosen according to the user's *contention window (CW)* variable. If it senses a busy channel during its back-off duration, it pauses its back-off timer and resumes the timer again at the end of the sensed transmission. A user transmits its packet only when its back-off timer expires. At the end of each successful transmission, the receiving user acknowledges the reception by sending an ACK message. Collisions occurs as a result of more than one user transmitting simultaneously. If a station does not receive an ACK, it doubles its contention window size up to a maximal CW value denoted by $CW_{max} = 2^K CW_{min}$, where CW_{min} is the minimum congestion window. The user resets its contention window size to its initial value, CW_{min}, after every successful transmission. When a user is successful in capturing the channel, it can then choose the data rate at which it wants to transmit the packet, preceded by a preamble that allows the receiver to lock on to and infer the parameters (e.g., rate, duration) of the transmission. The default values of these parameters for the IEEE 802.11a/b/g family of standards is given in Table I. Detailed description of the IEEE MAC protocol can be found in [14].

Parameter	802.11 b	802.11 a	802.11 g
CW_{min}	32	16	16
CW_{max}	256	256	256
SIFS	$10\mu s$	$16\mu s$	$10\mu s$
Slot time	$20\mu s$	$9\mu s$	$9\mu s$
DIFS	$50\mu s$	$34\mu s$	$28\mu s$
Preamble	$192\mu s$	$20\mu s$	$20 + 6\mu s^{\dagger}$
Max data rate	11 Mbps	54 Mbps	54 Mbps
Data symbol length	8 bits	27 Bytes	27 Bytes

TABLE I
MAC AND PHY PARAMETERS FOR IEEE 802.11 A/B/G

† 802.11g adds a $6\mu s$ "signal extension time" at the end of every frame.

We consider only the 802.11a standard for the experimental evaluation in this paper. The analysis and experimental con-

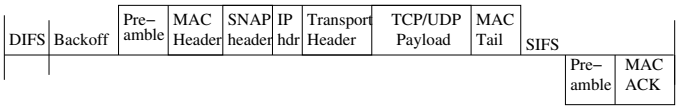

Fig. 1. 802.11 Transmit Sequence with Zero Collisions

clusions do apply for 802.11g as well, as long as the I-WLAN operates in 802.11g-only mode.

B. Network Model

An I-WLAN is a single cell system comprising of one AP and multiple users associated with that AP. Multiple such cells can co-exist within range of each other as long as they are on non-interfering channels. Within a cell, all traffic to users in the cell is routed through the AP (i.e., there is no direct user-user communication). Traffic is a mixture of TCP and UDP flows, with the data transfer skewed heavily to the downstream ($\geq 90\%$), than the upstream. This is typical of a hot-spot or home-network setting, where most users request and download web content, rather than produce their own. The upstream traffic is composed of content requests, audio/video conferencing streams, and file sharing streams. These upstream flows are mostly rate limited[1], or are composed of only a few packets.

Downlink TCP flows generate return traffic to the AP from users in the form of TCP ACKs. In general, the *deferred ACK* feature (set to 2, by default) in all TCP flavors ensures that we get only *far fewer TCP ACK packets* as that of TCP data packets, assuming in-order delivery. However, these TCP ACK packets must also contend with upstream flows and TCP ACK packets from other users in the cell, in addition to the packets from the AP.

III. Performance reduction due to idle channel time

We first consider the ideal performance of a single IEEE 802.11 cell with zero contention losses. We look at the best-case performance for UDP as well as TCP using the default settings of IEEE 802.11a, and the results are similar for 802.11g as well.

Figure 1 shows the sequence of events at the MAC layer assuming that there are no collisions in the cell. The 802.11 MAC header has 28 bytes and the SNAP encapsulation header has an additional 8 bytes. Maximum throughput is achieved when the packet size is the maximum. Since the Ethernet payload size is limited to 1500 bytes, we will use that number as the size of the MAC payload even if 802.11 MAC allows for a larger frame size. For an UDP packet, the maximum UDP payload length is 1472 bytes, and for TCP, it is 1460 bytes out of the maximum MAC frame size of 1536 bytes. The TCP ACK frame is 76 bytes in length. Each data symbol in 802.11a carries 27 bytes at the 54 Mbps data rate, and 3 bytes at the lowest 6 Mbps data rate.

[1]The default configurations in file sharing applications typically limit the uplink bandwidth[15] in order to increase download speeds.

The average number of the back-off slots is half of the CW variable. In the ideal case, $CW = CW_{min}$, and hence, the average back-off duration is 802.11a is $Slot\_time * CW_{min}/2 = 72\mu s$. The preamble duration is 20 μs. Thus, the total time needed to transmit a packet is

DIFS + Back-off + Preamble + Frame Duration + SIFS + Preamble + MAC ACK Duration

$$= 162 + 4\left(Data\_Symbols + MAC\_ACK\_Symbols\right)\mu s$$

Ideal max UDP throughput: In case of UDP, the packet duration of the maximum sized frame is

$$162 + 4(\lceil\frac{1536}{27}\rceil + \lceil\frac{14}{27}\rceil) = 394\mu s$$

This implies that the maximum UDP throughput is $\frac{1472\times 8}{394} = 29.9$ Mbps. It is important to note that the channel is idle for *DIFS + Back-off + SIFS* . With the average back-off time around $72\mu s$, the channel is idle for $122\mu s$, nearly 31 % of the time it takes to transmit one UDP frame.

Ideal max TCP throughput: In case of TCP, we have to account for the transmission duration of the TCP ACK packets as well. The only difference in case of the TCP ACK packet is that we do not have to account for back-off for the TCP ACK since both the AP and the user will be counting down the back-off counter. Since we already count the AP's back-off, we do not have to count it again. In addition, since TCP uses *deferred ACKs*, typically set to once every 2 packets, we compute the total duration of transmission of two TCP data packets and one TCP ACK packet as follows:

$$2\{394\} + \{90 + 4(\lceil\frac{76}{27}\rceil + \lceil\frac{14}{27}\rceil)\} = 894\mu s \qquad (1)$$

Thus, the expected duration of a TCP data frame is $894/2 = 447\mu s$, resulting in a maximum TCP throughput of $\frac{1460\times 8}{447} = 26.1$ Mbps. In this set of three packet exchanges, the channel is idle for 294 $\mu s = 33\%$ of the time. Note that the above computation does not take into account any potential collisions between the AP and the user sending a TCP ACK. The real throughput is therefore lower than 26 Mbps.

The above effects can be observed experimentally as well. We evaluate the throughput of users in a WLAN in infrastructure mode with only downlink flows, and compare the UDP and TCP throughput obtained with the above analytical results. The results are summarized in Table II. Throughput is in Mbps.

TABLE II
Single User Experiment: 54Mbps & 6 Mbps mode

CWmin		TCP		UDP	
		54Mbps	6Mbps	54Mbps	6Mbps
16	throughput	22.90	4.66	27.20	5.16
	(succ. prob)	(0.98)	(0.97)	(1.00)	(1.00)
32	throughput	20.50	4.60	23.10	4.97
	(succ. prob)	(0.99)	(0.98)	(1.00)	(1.00)

As is evident, even for the simplest case of a cell with only one user, after accounting for all data headers, the channel is idle for nearly one-third of the time. Out of this, back-offs account for 60% of the idle time, even when $CW_{min} = 16$. When the $CW_{min} = 32$, the channel is idle for 42% of the time, with back-offs accounting for 75% of the channel idle time. Conventional wisdom has been to use higher values of CW_{min} to resolve collisions that are likely when a large number of users contend for the channel[3], [9]. While this definitely will reduce the impact of collisions, it also increases the amount of idle time on the channel, especially with high data-rate WLANs, and will be further exacerbated in 802.11n WLANs. At low data rates, the channel idle time is less pronounced ($< 5\%$). However, the biggest benefit of using the high-rate WLAN standards is the availability of high-data rates, and their impact is dulled by the overhead of back-offs and channel idle time. Moreover, when there are multiple flows, there can be additional contention among users and with the AP as well, which will lead to collisions, doubling of the back-off window and subsequent loss in throughput.

Our goal is to show that for the traditional and most common single-cell WLAN setting, it is possible to achieve better throughput by reducing the back-off window without increasing the risk of collisions. Having said that, the answer is obviously not to eliminate back-off, but to find the right balance between improved contention resolution and reduced channel idle times. We seek to find this balance by means of analysis, and then use our analytical results in experiments to validate our conclusions.

IV. Performance Analysis

In this section, we present our analytical model evaluating the performance of a single AP that provides multiple TCP and UDP downstream flows to its associated users. We introduce a new Markov model that evaluates the system performance from the AP's point of view and use this model for computing the AP's success probability and its overall throughput. Since the stations' behavior in WLANs is not generally Markov in nature, we make several simplifying assumptions that enable us to provide a tractable model. Yet, our extensive experiments show that in spite of its simplicity, our model predicts quite accurately the performance of the system.

A. Simplifying Assumptions

We consider a WLAN with N downstream TCP and UDP flows, each flow is sent to a different user. For TCP flows we assume that the users apply delayed TCP ACK (TACK) strategy and TACK are sent only after reception of several TCP messages, typically this value is $D = 2$. In order to preserve the Markov model, we assume that after reception of a correct TCP frame, a user decides with probability of $1/D$ whether to send a TACK. Thus, while at any given time there may be at most N users that have TACK messages to send to the AP, this number may actually be significantly smaller. Once a user successfully sends its TACK message to the AP, it no longer competes for the channel until it gets at least one new TCP message and the number of users with pending TACK messages is increased by one. We assume that the only cause of getting a corrupted TCP frame is due to packet collision at the WLAN DCF layer.

Since, our main focus is the AP performance, we assume that all the users have a fixed contention window denoted by U that does not change even in case of a collision. Our experiments support this assumption and show that moderate variation of U does not have significant impact on the system performance. Thus in our Markov model, we assume that after each transmission, the probability that a user is contending in slot $j \leq U$ is $1/U$, which is comparable to the assumption made by Bianchi in [16]. For the sake of clarity, we present the notations used in Table III.

Symbol	Semantics
W	The minimal contention window (CW) of the AP.
U	The contention window of the users.
D	The ratio of TCP frames to TCP ACKs.
K	Num. of CW duplications, *i.e.,* $CW_{max} = CW_{min} \cdot 2^K$.
S_i (n_i, k_i)	A state at the Markov Chain, where n_i is the number of TACKs and k_i is number of AP CW duplications.
$P(n_i, k_i)$	the probability of state $S_i = (n_i, k_i)$.
$T(S_i; S_j)$	The transition probability from state S_i to S_j.
$m(S_i, r)$	The probability of transmitting r successful TACKs before the AP transmitted its message in state S_i.
$A(S_i)$	The probability of successful AP transmission in state S_i.

TABLE III
Notation

B. Markov Model

In the IEEE 802.11 back-off model, when the AP does not receive an acknowledgement for its transmission, it doubles its back-off window, up to a maximum value, $CW_{max} = 2^K W$, for some K. Upon a successful transmission, indicated by an ACK, it resets its back-off window to W. We model the transmission of the AP by a two-dimensional Markov chain in which each state is defined by a pair (n, k). Here, n denotes the number of users that have pending TACKs to send, while k represents the AP CW after i consecutive collisions, i.e., the AP CW is $W \cdot 2^k$. Thus, the Markov chain represents the system state from the AP point of view immediately after it has sent a message.

We denote the states of the Markov chain by a pair $S = (n, k)$ and the probability that the AP is in state (n, k) is given by $P(n, k)$. The state transition probabilities from state $S_i = (n_i, k_i)$ to $S_j = (n_j, k_j)$ are denoted by $T(S_i; S_j)$. Each state transition probability takes into account two probabilities: (i) The probability that the message sent by the AP has been received correctly and acknowledged by the users. This probability is denoted by $A(s_i)$ and it is calculated in Section IV-C. In the case of successful transmission, the AP resets its CW to W and it moves to state $S_j = (n_j, 0)$, with $k_j = 0$. Otherwise, in case of a collision, the AP doubles its CW and moves to state $S_j = (n_j, k_i + 1)$, *i.e.,* $k_j = k_i + 1$. If $k_i = K$ (maximal contention window), then k_j is the same as $k_i = K$.
(ii) The probability that the number of users with pending

TACK messages has changed from n_i to n_j. This depends on the successful delivery of data from the AP (which increases the pending users) and the number of users that transmitted TACK messages successfully prior to the AP transmission (which decreases the number of pending users). We denote by $m(S_i, r) = m(n_i, k_i, r)$ the probability that r users have successfully sent their TACK messages before the AP transmitted its message.

Given a state (n_i, k_i) its transition probability can be calculated as follows. In case of an AP transmission failure, $k_j = \min(k_i + 1, K)$ and the probability of $n_j = n_i - r$ users with pending TACK messages is $m(S_i, r) = m(n_i, k_i, r)$. From this, the transition probability for unsuccessful transmission is given by

$$T(n_i, k_i; n_j, k_j) = [1 - A(n_i, k_i)] \cdot m(n_i, k_i, n_i - n_j)$$

$$0 \leq n_j < n_i, \qquad k_j = min(k_i + 1, K)$$

Recall that in such a situation, $1 \leq n_j \leq n_i$, since the AP has collided with at least one user and also since no new TACK was generated (i.e., the downlink TCP data transmission failed).

The case when the AP successfully transmits is slightly more complicated, since the receiving user generates a new TACK only with probability of $1/D$. We distinguish between three possible scenarios.

The first is where $n_j = n_i + 1$. Here, a new TACK is generated and none of the users successfully sent any pending TACK message. The probability for such case is;

$$T(n_i, k_i; n_i + 1, 0) = A(n_i, k_i) \cdot \frac{1}{D} \cdot m(n_i, k_i, 0)$$

The second case is the transition to the state $(0, 0)$. In this case, all the pending TACK are successfully transmitted and no new TACK is generated. The probability of this case is

$$T(n_i, k_i; 0, 0) = A(n_i, k_i) \cdot \frac{D-1}{D} \cdot m(n_i, k_i, n_i)$$

Finally, the third case is when $n_j \in \{1, \cdots, n_i\}$. In such case the transition probability is,

$$T(n_i, k_i; n_j, 0) = A(n_i, k_i) \cdot \left[\frac{D-1}{D} \cdot m(n_i, k_i, n_i - n_j) \right.$$

$$\left. + \frac{1}{D} \cdot m(n_i, k_i, n_i + 1 - n_j) \right]$$

Inside the parenthesis, the first component considers the case that the receiving user did not generate a new TACK. The second component evaluates the probability that new TACKs were generated. Here, the system shifts to state $(n_j, 0)$ only if $n_i - n_j + 1$ TACK messages have been successfully sent. Figure 2 provides a diagram of our Markov model that shows all possible out-going transitions from state (n, k).

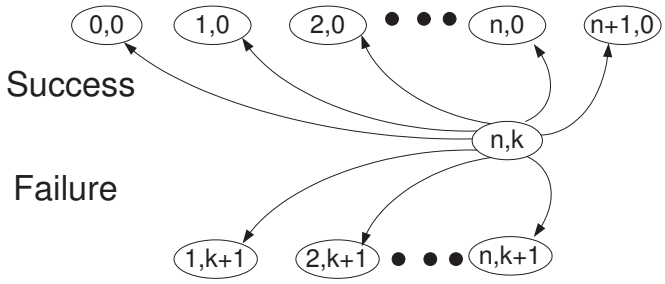

Fig. 2. An example of the out going transitions from the state (n, k).

C. Probability of Successful AP Transmission - $A(S_i)$

We now calculate the probability of successful AP transmission in state $S_i = (n_i, k_i)$. Denote $\hat{W} = W \cdot 2^{k_i}$. We consider two cases.

Case I: $\hat{W} \leq U$. The probability that the given user selects a different slot from the one selected by the AP is $\frac{U-1}{U}$. Thus, with n_i pending users, the success probability is given by

$$A(n_i, k_i) = \left(\frac{U-1}{U} \right)^{n_i}$$

Case II: $U < \hat{W}$. If the AP selects a slot with index $s > U$ then there is no collision, since all the TACKs have already been transmitted. The probability of such event is $\frac{\hat{W}-U}{\hat{W}}$. Collision may occur only if the AP selects a slot $s \leq U$, with the probability of collision being $1/U$. Thus the success probability is;

$$A(n_i, k_i) = \frac{\hat{W} - U}{\hat{W}} + \frac{U}{\hat{W}} \cdot \left(\frac{U-1}{U} \right)^{n_i}$$

D. Probability of Successful TACK Transmissions - $m(n_i, k_i, r)$

Given a state $S_i = (n_i, k_i)$, we first calculate the probability of r users transmitting their TACK before the AP transmits its message and then use this probability to estimate the number of successful TACK transmissions. This simplifying assumption is reasonable since the the number of collisions between TACKs is relatively small. On one hand, when the user CW, U, is small relative to W, then the users attempt to transmit their TACKs before the AP transmits. Thus, the number of contending users is very small and it is typically no more than one. For example, for the case of $W = 8$ and $U = 4$ (a relatively worst-case scenario), the computed probability of more than 1 user contending with the AP is only 0.097, and the probability of more than 2 users contending with the AP is 0.0039.

Based on this, we can assume that the probability of collision between TACKs is small. On the other hand, when $W < U$ the number of contending users is high, however, they have a large contention window leading to a low probability of collision. We now calculate the probability of r out of n_i users transmitting their TACKs before the AP's transmission, by providing a recursive calculation of $m(n_i, k_i, r)$.

We select r of the n_i users and calculate the probability that the selected r user choose a transmission slot smaller than the slot s selected by the AP, while the other $n_i - r$ users choose a slot with index the same or higher than s. The first probability is given by $m(r, k_i, r)$ while the second probability is given by $m(r, k_j, 0)$. It follows that,

$$m(n_i, k_i, r) = \binom{n_i}{r} \cdot m(r, k_i, r) \cdot m(r, k_i, 0)$$

For calculating the basis of our recursive formulation, we distinguish between two cases.

Case I: $\hat{W} \leq U$. In this case the probability, Y_I, that a given user selects a slot j smaller than s is as follows.

$$Y_I = \sum_{s=2}^{\hat{W}} \sum_{j=1}^{s-1} \frac{1}{U \cdot \hat{W}} = \sum_{s=2}^{\hat{W}} \frac{s-1}{2 \cdot U \cdot \hat{W}} = \frac{\hat{W}-1}{2 \cdot U}$$

Therefore, we get

$$m_I(r, k_i, r) = \left(\frac{W \cdot 2^{k_i} - 1}{2 \cdot U} \right)^r$$

$$m_I(r, k_i, 0) = \left(1 - \frac{W \cdot 2^{k_i} - 1}{2 \cdot U} \right)^r$$

Case II: $U < \hat{W}$. As we calculated earlier for $A(S_i)$, with probability $\frac{\hat{W}-U}{\hat{W}}$ the AP selects a slots $s > U$, and then the r users transmit their TACKs before the AP's transmission. We also need to calculate the probability of such an event when $s \leq U$. The probability that a given user chooses a slot $j < s$ is given by,

$$\sum_{s=2}^{U} \frac{1}{U} \cdot \sum_{j=1}^{s-1} \frac{1}{U} = \frac{1}{U^2} \sum_{s=2}^{U} s - 1 = \frac{U-1}{2 \cdot U}$$

Thus, the probability Y_{II} that a given user selects a slot j smaller than s is as follows;

$$Y_{II} = \frac{\hat{W}-U}{\hat{W}} + \frac{U}{\hat{W}} \cdot \frac{U-1}{2 \cdot U}$$

By using similar argument for the case on r contending users we get,

$$m_{II}(r, k_i, r) = \frac{W \cdot 2^{k_i} - U}{W \cdot 2^{k_i}} + \frac{1}{W \cdot 2^{k_i} \cdot U^r} \sum_{s=2}^{U} s^r$$

$$m_{II}(r, k_i, 0) = \frac{1}{W \cdot 2^{k_i} \cdot U^r} \cdot \sum_{s=1}^{U} (U - s + 1)^r$$

E. Success Probability and Throughput

The Markov chain presented above enables us to calculate iteratively the probability $P(S_i) = P(n_i, k_i)$ that the AP is located in state $S_i = (n_i, k_i)$. From this we can calculate the success probability as

$$Pr(success) = \sum_{S_i} P(S_i) \cdot A(S_i)$$

For comparison with our experiment results we calculated the retry-rate, R;

$$R = \frac{Pr(failure)}{1 + Pr(failure)} = \frac{1 - Pr(success)}{2 - Pr(success)} \qquad (2)$$

For calculating the system throughput, we calculate the expected amount of information delivered by a single message as $Pr(success) \cdot msg\_payload$ divided by $Avr\_overall\_msg\_time$, the time spent for transmitting a single TCP message including all the related overheads. In other words,

$$Throughput = \frac{Pr(success) \cdot msg\_payload}{Avr\_overall\_msg\_time}$$

We now elaborate on our $Avr\_overall\_msg\_time$ calculation. This period of time contains both fixed and varying time period components that are included in the overall period of time that each message transmission consumes. The fixed time-period components include the DIFS and the packet transmission duration itself. In addition, we take into account the following varying time-period components:

(a) The average contention window, denoted by avr_CW,

$$Avr\_CW = \sum_{S_i = (n_i, k_i)} P(n_i, k_i) \cdot \frac{W \cdot 2^{k_i} - 1}{2} \cdot Slot\_time$$

(b) SIFS and ACK message in case of successful transmission, (c) TACK message transmission for every D successful delivery of TCP messages, where $D = 2$ for considering delayed TACK transmission, and (d) the overhead of TACK collision. Due to lack of space, we do not present the details of the last component in this paper. As noted earlier in this section, the probability of TACK collision is very small.

Recall that when the AP support both TCP and UDP flows, the ratio D of data messages to TACK messages should be increased to meet the ratio between TCP flows and UDP flows. For instance, if half of the flows are TCP, while the other flows are UDP, then under the assumption of fair bandwidth allocation to each flow, the value of D is 4. From this, it follows that,

$$Avr\_overall\_msg\_time = avr\_CW + msg\_Tx\_time$$

$$+DIFS + Pr(success) \cdot [SIFS + ACK\_time +$$

$$+\frac{1}{D} \cdot (DIFS + TACK\_time + SIFS + ACK\_time)]$$

Recall that the overhead of transmitting TACK messages does not contains CW time, since this time overlaps with the AP contention window duration.

F. Modeling UDP traffic

Our proposed model is tailored to the I-WLAN setting with dominant downstream traffic. It can also be extended to the case of UDP flows as well as moderate TCP/UDP upstream flows. This approximation to the analysis is primarily done via the *delayed ACK* parameter, D.

The parameter D can be interpreted in a couple of ways: (a) it indicates how many packets are sent on the downlink in order to generate a single packet on the uplink, or (b) it is the ratio of downlink to uplink traffic. For the case of a TCP downlink flows alone, a single TCP-ACK message is sent for every successful transmission of $D = 2$ TCP downstream packets.

If the system supports a small number of upstream flows, D may get a value $1 \leq D \leq 2$ for representing the upstream traffic. For instance, consider the case of x downlink TCP flows and y upstream TCP flows, where $y = c \cdot x$, for some $0 \leq c \leq 1$. All TCP flows have deferred TCP ACK ratio of 2 and we assume fair service to all flows. In such settings, for every successfully reception of $2x$ downlink TCP packets x TCP ACK message are sent upstream. Similarly for every $2y$ successful transmission of upstream TCP packets, y downlink TCP ACK are sent. Consequently, under the assumption of fair service, D can be calculated as follows, $D = \frac{\text{Downlink}}{\text{Upstream}} = \frac{2x+y}{x+2y} = \frac{2+c}{1+2c}$. For $c = 0, 0.25, 1$, $D = 2, 1.5, 1$ respectively.

The analysis for UDP flows is along similar lines, as long as the assumption of fair service among upstream flows is maintained. If we have UDP downlink flows with a rate of k_d packets and UDP upstream flows at the rate of k_u packets for every downlink TCP packet (in the equilibrium state), then we can model this case by setting $D = \frac{2+c+2k_d}{1+2c+2k_u}$, as long as $k_u \leq k_d$.

G. The timing factor

We evaluated the performance of our analytical model via an extensive set of experiments and we observed a departure from the analytical model when the user contention window is set to 2. For instance, in the case when the AP CW, $W = 2$, and the user CW, $U = 2$, we expected a success probability of around 65%, while in practice we detected success probability of $75-81\%$. This performance gap cannot be easily explained for reasons described next.

Consider for instance the case of a single TCP follow where $W = 2$, $U = 2$ and the ratio of TCP frames to TCP ACKs is $D = 2$. When both the user and the AP have a packet to send there is a probability of 50% that their transmission will collide. Recall that the user produces a TCP ACK packet after reception of two TCP frames, which implies the overall failure probability of at least 25%. When we take into account the non-negligible probability that the retry messages may collide as well, this implies that the success probability should be significantly lower than 75% and it should be around 65%. We rule out estimation error as the cause since our model produces quite accurate estimates that match the experimental data for higher contention window values. Thus, it appears that *even when the user and the AP select the same slot their transmissions may not collide*. We believe that such phenomenon is entirely possible due to timing issues that can be explained as follows.

Consider an AP with a single user and let us assume that the AP has just finished delivery of a message to the user. Let us say that the user has a packet to send to the AP, and now both AP and user select the same back-off slot. Both the user and AP sense idle channel, wait for the same amount of back-off slots after both wait for the DIFS time. Now, note that since the AP's last transmission was a success, the last message seen on the channel is the MAC ACK sent by the user to the AP. This implies that the user's radio is in Tx mode, and it should now switch to Rx mode, while the AP is already in Rx mode[2]. If a node decides to transmit at the end of the back-off + DIFS period, it should switch to Tx mode. For the transmission to be successful, the message should reach the other end for at least T_{CCA} (Clear Channel Assessment Time) before the other node switches to Tx mode. In other words,

$$T_{Tx \to Rx} > T_{Rx \to Tx} + T_{CCA}$$

where, $T_{Tx \to Rx}$ and $T_{Rx \to Tx}$ denotes the transition time of switching the radio from Tx to Rx mode and vice versa, accordingly. Therefore, in this scenario, if the two nodes select the same back-off window then the last node to send a transmission on the air (here, a MAC ACK) will detect the other node's subsequent transmission first and defer its own transmission. For IEEE 802.11a/g, these radio Tx/Rx mode transition times are $< 5\mu s$, while the T_{CCA} time is approximately $2\mu s$. Depending on the specific vendor chipset and the hardware timing inaccuracy, there can be a non-negligible probability of success even when two nodes select the same slot.

In order to evaluate this theory, we conducted two experiments in different testbeds. In the first testbed, there is one AP and a MN is attached to this AP. The MN has one downstream UDP flow and one upstream UDP flow. The UDP payload size is set to 1472 bytes. The CW_{min} of AP and MNs are set to either 2 or 4. To ensure the same back-off behavior, we set CW_{max} to the same value as CW_{min}. In the second testbed, there are two APs (AP1 and AP2) and two MNs (MN1 and MN2). Both APs use the same frequency. MN1 is attached AP1 while MN2 is attached to AP2. Each MNs has a downstream UDP flow with payload size of 1472 bytes. The experimental results are shown in Table IV. The main observation is that in the one AP setup, the upstream flow receives clearly higher throughput than the downstream flow, while such unfairness does not occur in the two AP setup. Conceptually in both cases there are always two stations that compete for the channel, and hence we expect them to behave identically. However, based on our earlier inference, the skew in the throughput in the single AP case is predictable.

In the two-AP case, the APs send UDP packets and the users reply with ACK packets, ensuring both AP simultaneously sense the idle channel and initiate their back-off mechanism at the same time. Thus their chance of collision is exactly the same. However, in the one-AP case, the user and AP are both the source and the destinations of the flows. Thus they toggle between sending UDP frames and ACK messages. This means that in this experiment after each successful transmission one

[2]Channel propagation times are negligible at these ranges, so we ignore them here.

TABLE IV
EXPERIMENT RESULTS: TIMING FACTOR

		1 AP & 1 MN		2 AP & 2 MNs	
		up	down	down_1	down_2
CWmin=2	throughput	16.00	11.70	14.10	13.50
	(succ. prob)	(0.50)	(0.32)	(0.41)	(0.36)
CWmin=4	throughput	18.30	11.80	14.80	14.70
	(succ. prob)	(0.75)	(0.60)	(0.64)	(0.62)

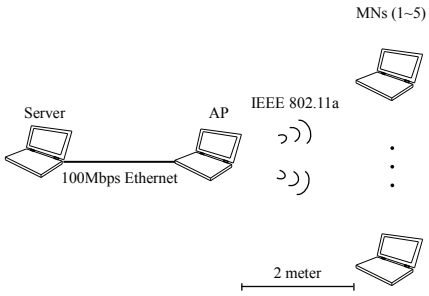

Fig. 3. Testbed layout

of the stations detects that the channel is idle some time after the other station. An initial throughput skew can easily build up as time progresses as the last successful UDP sender continues to succeed most of the time when a collision is supposed to happen. This experiment confirms our timing gap theory and we incorporate it in our model. Essentially, in the case of a collision, we evaluate the probability that one of the station was the last station that sent an ACK packet and in such case we assume a 25% probability that the stations do not collide. This modification enables us to provide a more accurate estimation of the system performance.

V. EXPERIMENTAL EVALUATION

The experiments were carried out in an indoor testbed with a single 802.11a cell. The testbed is depicted in Figure 3. All devices, including WLAN AP, in the testbed are Linux-based notebook computers (Thinkpad z60m) to achieve maximum flexibility. All notebooks commonly run *Fedora core 5* (2.6.15.1 kernel). In that kernel, every TCP ACK is delayed 200 msec ($D = 2$) as default. For WLAN interface, Cisco Aironet 802.11 a/b/g adapter (with Atheros chipset) and 'madwifi-0.9.4' device driver are used. We use the *Iperf* for the traffic generation. In all experiments, 802.11a is used at channel 34 (5.17GHz). There is no external interference on this channel. The server is connected to the AP via 100Mbps Ethernet. To make the testbed more realistic, we run *Netem* delay emulator between the server and the AP. In all experiments, 20 msec artificial delay is injected for the traffic between the server and the AP. The MNs are located 2 meters away from the AP to ensure the channel condition for 54Mbps transmission mode. MNs are positioned about 1 meter apart from each other to avoid interference. We collocate all MNs in order to avoid any skewing of results due to the well-known location-based unfairness observed in WLANs due to capture effects [18]. Since in our work we just tune the contention window of the AP and the mobile nodes, note that the performance trends shown here are independent of the nodes' position and their bit-rates. Each experiment is run for 6 minutes. The first 3 minutes is for warm up and the data measurement is done during the last 3 minutes.

A. Smaller is better

We first examine the impact of CW_{min} selection when there exist only TCP downstream flows. Five MNs, each of which runs 3 TCP downstream flows, are attached to the AP. A total of 25 possible combinations of CW_{min} setting are tested. Five different values (which are 2, 4, 8, 16, and 32) are used as the CW_{min} of AP and the same five values are used as the CW_{min} of MNs. All MNs use the same CW_{min} in each experiment. In each experiment, we measured the TCP throughput for each MN, the total TCP throughput which is the sum of individual MN's throughput, and retry rate. The TCP throughput is measured at the MN side, and the retry rate is measured at the AP side. Table V summarizes the results. The TCP throughput of individual MNs is omitted as all MNs share the total throughput roughly equally in all cases. Instead of the retry rate, we present the success probability (using Equation (2)).

The first general observation from Table V is that smaller CW_{min} generally increases the system throughput. For example, the case of (CW_{min} of AP = 16, CW_{min} of MN = 16) results about 14.2% higher throughput than the case of (CW_{min} of AP = 32, CW_{min} of MN = 32). The case of (8, 8) produces 20.3% higher throughput than the case of (32, 32). Reducing AP's CW_{min} to below 8, however, causes throughput degradation. For MN's CW_{min}, such a point of inflection does not appear, and using smaller CW_{min} is nearly always beneficial. As a result, the case of (8, 2) becomes the optimal CW_{min} setting in this experiment. Note that the throughput is the outcome of the interplay between the success probability and CW_{min} value, so that higher success probability does not warrant higher throughput. The comparison of experimental data and analytic data indicate that our model predicts the experimental throughput remarkably well.

B. Contention is constant

One of the key results that we observe from our analysis is that the throughput is largely independent of the number of TCP flows, confirming the result noted in [11]. To verify this by experiment, we changed the number of TCP flows per MN to 1, 2, and 3, resulting a total of 5, 10, and 15 TCP flows in the system. The results are given in Table VI, where analytic results are given in the last column as they are not affected by the number of flows. For the sake of clarity, we only present the CW_{min} settings which use the same value for AP and MNs, i.e., (2,2), (4,4), (8,8), (16, 16), (32, 32). We can observe that the throughput is largely unaffected by the number of TCP flows. We also conducted experiments by changing the number of MNs, and witnessed identical behavior.

TABLE V
TCP DOWNLINK FLOWS ONLY (THROUGHPUT IN MBPS)

Experiment			MN CWmin (U)				
			2	4	8	16	32
AP CWmin (W)	2	throughput	23.18	22.79	22.42	22.18	22.49
		(succ. prob)	(0.75)	(0.72)	(0.70)	(0.68)	(0.69)
	4	throughput	23.77	23.64	23.51	23.37	23.27
		(succ. prob)	(0.82)	(0.81)	(0.81)	(0.80)	(0.79)
	8	throughput	24.36	24.32	24.18	24.11	23.94
		(succ. prob)	(0.92)	(0.92)	(0.91)	(0.91)	(0.90)
	16	throughput	22.99	23.02	23.02	22.95	22.94
		(succ. prob)	(0.96)	(0.96)	(0.96)	(0.96)	(0.96)
	32	throughput	20.17	20.16	20.17	20.16	20.09
		(succ. prob)	(0.99)	(0.99)	(0.99)	(0.98)	(0.98)

Analysis			MN CWmin (U)				
			2	4	8	16	32
AP CWmin (W)	2	throughput	22.12	22.08	22.13	22.12	22.39
		(succ. prob)	(0.72)	(0.72)	(0.72)	(0.72)	(0.73)
	4	throughput	25.40	24.25	24.17	24.11	24.07
		(succ. prob)	(0.87)	(0.83)	(0.83)	(0.83)	(0.82)
	8	throughput	25.73	25.52	24.77	24.68	24.61
		(succ. prob)	(0.94)	(0.94)	(0.91)	(0.90)	(0.90)
	16	throughput	23.96	23.95	23.84	23.41	23.35
		(succ. prob)	(0.97)	(0.97)	(0.97)	(0.95)	(0.95)
	32	throughput	20.46	20.48	20.47	20.41	20.19
		(succ. prob)	(0.99)	(0.99)	(0.99)	(0.98)	(0.97)

TABLE VI
TCP DOWNLINK ONLY (VARYING FLOW COUNTS)

(W, U)		5 flows	10 flows	15 flows	Analysis
(2,2)	throughput	24.65	23.95	23.18	22.12
	(succ. prob)	(0.81)	(0.78)	(0.75)	(0.72)
(4,4)	throughput	24.67	24.20	23.64	24.25
	(succ. prob)	(0.85)	(0.83)	(0.81)	(0.83)
(8,8)	throughput	25.01	24.58	24.18	24.77
	(succ. prob)	(0.93)	(0.92)	(0.91)	(0.91)
(16,16)	throughput	23.53	23.25	22.95	23.41
	(succ. prob)	(0.97)	(0.97)	(0.96)	(0.95)
(32,32)	throughput	20.52	20.35	20.09	20.19
	(succ. prob)	(0.99)	(0.99)	(0.98)	(0.98)

C. UDP improves overall throughput

We then examine the scenario when TCP downstream flows co-exist with UDP downstream flows. The same five MN testbed is used. To assess the impact of the UDP downstream traffic, each MN always runs one TCP downstream flow while we tested two cases, one without UDP downstream flow, one with UDP downstream flows. To prevent UDP traffic from dominating the TCP traffic, we used rate-limited UDP flows (to 500Kbps in this experiment). 6 UDP downstream flows were added for each MN. Table VII shows the results. Only the results of some important CW_{min} settings are presented. The key observation is that introducing UDP downstream traffic results in higher success probability and higher total throughput, and as noted in Table V, the (8,2) setting for the AP and User CW works out best.

TABLE VII
EXPERIMENT AND ANALYSIS RESULTS: TCP DOWNLINK + UDP DOWNLINK FLOWS

(W, U)		TCP only		TCP + UDP	
		Experiment	Analysis	Experiment	Analysis
(4,2)	throughput	24.83	25.41	26.39	26.73
	(succ. prob)	(0.86)	(0.88)	(0.91)	(0.92)
(8,2)	throughput	25.12	25.73	25.03	26.41
	(succ. prob)	(0.94)	(0.94)	(0.95)	(0.97)
(8,4)	throughput	25.05	25.52	24.95	26.26
	(succ. prob)	(0.94)	(0.94)	(0.95)	(0.96)
(16,16)	throughput	23.49	23.41	22.74	23.97
	(succ. prob)	(0.97)	(0.95)	(0.98)	(0.97)

D. Large CW is better for upstream flows

Another interesting scenario is when TCP downstream flows are mixed with TCP upstream flows. We again used the five MN tested, in which each MN runs either only TCP downstream flows or only TCP upstream flows. The number of flows per MN does not influence the result. We experiment with three different configurations by the changing the number of MNs with upstream flows, from one MN to three with remaining MNs running downstream flows. In this experiment, we fix the CW_{min} of AP by 8, while applying different values for the CW_{min} of MN with upstream flows (i.e., 16 and 32) and the CW_{min} of MN with downstream flows (i.e., 2, 4, and 8). The experimental result is summarized in Table VIII.

When the CW_{min} of MN with upstream flows is set to 16, upstream MNs receive more than their fair share of throughput at the expense of downstream MNs. For example, when the CW_{min} of downstream MNs is 2, one upstream MNs get 8.79 Mbps while 4 downstream MNs get only 15.72 Mbps (each gets 3.93 Mbps). When the CW_{min} of MN with upstream flows is increased to 32, such unfairness reduces significantly, while total system throughput is virtually unaffected. From this observation, we conclude that MNs with TCP upstream flows should use high CW_{min} values, even bigger than 32. In contrast, the MNs with TCP downstream flows should use small CW_{min} values, such as 2, 4, or 8. As the portion of upstream MNs increases, the unfairness between upstream MNs and downstream MNs reduces noticeably, but the overall system throughput also drops. Independent of the mixture pattern of upstream MNs and downstream MNs, using large CW_{min} for upstream MNs and using small CW_{min} for downstream MNs always provides better performance. In this experiment, the analytical model is able to predict the success probability of the AP precisely. However, since we do not model the throughput of upstream TCP flows, the model cannot accurately predict the system throughput, unlike the downlink-only cases seen before.

E. Recommendations for setting CW

As is evident from the results above, our experimental evaluation agrees with our analytical model, and provides us with four key conclusions. The first three are as follows.

TABLE VIII
EXPERIMENTAL RESULTS: TCP DOWNLINK FLOWS + TCP UPLINK FLOWS (AP'S $CW_{min} = 8$)

(up MN CWmin, down MN CWmin)		1 up MN + 4 down MNs		2 up MNs + 3 down MNs		3 up MNs + 2 down MNs	
		up	down	up	down	up	down
(16,2)	throughput	8.79	15.72	14.87	8.67	18.45	4.11
	(succ. prob)	(0.72)	(0.85)	(0.64)	(0.77)	(0.55)	(0.69)
(16,4)	throughput	8.80	15.42	14.81	8.68	18.34	4.11
	(succ. prob)	(0.72)	(0.85)	(0.64)	(0.77)	(0.54)	(0.68)
(16,8)	throughput	8.76	15.14	14.80	8.65	18.60	4.51
	(succ. prob)	(0.67)	(0.84)	(0.64)	(0.77)	(0.57)	(0.68)
(32,2)	throughput	6.24	17.97	13.05	10.66	15.07	6.93
	(succ. prob)	(0.65)	(0.87)	(0.65)	(0.80)	(0.58)	(0.74)
(32,4)	throughput	6.18	17.97	12.99	10.67	16.62	7.23
	(succ. prob)	(0.64)	(0.87)	(0.65)	(0.79)	(0.57)	(0.74)
(32,8)	throughput	6.29	17.88	12.89	10.74	15.84	6.92
	(succ. prob)	(0.67)	(0.87)	(0.64)	(0.79)	(0.55)	(0.73)

(a) CW_{min} for the AP should be set to 8, (b) CW_{min} for upstream TCP ACK traffic should be set[3] to 4, (c) CW_{min} for other upstream TCP traffic at the MNs should be set to at least 32 to ensure fairness for downstream flows, if the MNs do not know the number of flows in the I-WLAN.

If the MNs in the I-WLAN have consistent information about the total number of flows in the cell, then further throughput optimization is feasible. When the number of upstream flows is large, then the CW_{min} for other upstream TCP traffic at the MNs should be set to a value greater than 32, in order to avoid unfairness and starvation at the AP. We leave it for future work to find the optimal values of the CW_{min} for various number of upstream flows. We also suggest to implement a signaling protocol that enables the AP to configure the CW_{min} value of the MNs according to the number of upstream and downlink flows observed by the AP. Note that the CW_{min} at the AP has to be kept small at all times in order to give it higher priority to access the channel.

VI. DISCUSSION

In this paper, we presented an analytical model for capturing TCP+UDP flow behavior in I-WLANs. Our analytical model is corroborated by experimental data which also reveals some interesting observations. The key among these is the observation that the minimum congestion window can be reduced below the 802.11a/g default value of 16. In addition, we also affirm that the AP is a special node regardless of the number of contending users in the cell, and that setting its CW = 8, while increasing the CW of other upstream users to 32 is beneficial than setting everyone's CW to 32. This is contrary to the conclusions in [3], [6], [9]. The last observation is that by giving higher priority to TCP ACK traffic, the overall throughput of the system can be increased. While this might seem obvious from a high level, we provide analytical and experimental justification for choosing such a policy, unlike the results in [13].

[3]TCP ACKs need higher priority.

Note that when packets are not maximum-sized, especially on the uplink, the overhead of back-off becomes a greater proportion of the channel usage. Packet size is not set to MSS only when it is the last packet of a stream, or if the flow is part of a CBR flow (such as VoIP). In these cases, the impact of our solution will be much greater since we reduce the back-off overhead significantly. The unfairness between TCP downlink and uplink flows seen in Table VIII is a well-known issue, and can be addressed in parallel using the dual-queue mechanism in [17]. This does not affect our analysis of the AP's overall success probability. We plan to expand our analysis to cover the upstream + downstream cases next. We also plan to evaluate the impact of TCP and UDP flows in very large I-WLANs as part of future work.

ACKNOWLEDGMENTS

This work has been supported by IT R&D program of MKE/KEIT [KI002137, Ultra Small Cell Based Autonomic Wireless Network]

REFERENCES

[1] IEEE Computer Society LAN MAN Standards Committee, "*Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*", November 1999.
[2] IEEE Computer Society LAN MAN Standards Committee, "*IEEE P802.11n/D9.00, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 5: Enhancements for Higher Throughput*", March 2009.
[3] 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, no. 6, pp. 785-790, Dec. 2000.
[4] F. Cali,M. Conti, and E. Gregori, "*IEEE 802. 11 Protocol: Design and Performance Evaluation of an Adaptive Backoff Mechanism*", IEEE J. Selected Areas in Comm., vol. 18, no. 19, pp. 1774-1786, Sept. 2000.
[5] Z. J. Haas and J. Deng, "*On optimizing the back-off interval for random access schemes*", IEEE Trans. Commun., vol. 51, no. 12, pp. 2081-2090, Dec. 2003.
[6] H. Ma, H. Li, P. Zhang, S. Luo, C. Yuan, and X. Li, "*Dynamic Optimization of IEEE 802.11 CSMA/CA Based on the Number of Competing Stations*", Proc. ICC04, Paris, France, June 2004.
[7] F. Daneshgaran, M. Laddomada, F. Mesiti, M. Mondin, and M. Zanolo, *Saturation throughput analysis of IEEE 802.11 in presence of non-ideal transmission channel and capture effects*, *IEEE Transactions on Communications*, 2008.
[8] F. Daneshgaran, M. Laddomada, F. Mesiti, M. Mondin, and M. Zanolo, *Unsaturated throughput analysis of IEEE 802.11 in presence of non-ideal transmission channel and capture effects*, *IEEE Transactions on Communications*, 2008.
[9] A. Ksentini, et. al., "*Deterministic Contention Window Algorithm for IEEE 802.11*", *IEEE PIMRC*, Sept. 2005.
[10] Atheros Communications, *http://www.atheros.com*.
[11] R. Bruno, M. Conti, and E. Gregori, "*Analytical modeling of TCP clients in Wi-Fi hot spot networks*", in Proceedings of IFIP Networking, 2004.
[12] IEEE Computer Society LAN MAN Standards Committee, "*IEEE Standard 802.11e: Amendment to IEEE Std. 802.11: Medium Access Control (MAC) Quality of Service Enhancements*", November, 2005.
[13] D. J. Leith, et. al., "TCP Fairness in 802.11e WLANs", *IEEE Communications Letters*, 9 (12), December 2005.
[14] Bob O'Hara and Al Petrick, *The IEEE 802.11 Handbook: A Designer's Companion, Second Edition*, Standards Information Network, IEEE press, 2005.
[15] Bittorrent, *http://www.bittorrent.com*.
[16] G. Bianchi, *Performance analysis of the IEEE 802.11 distributed coordination function*, *IEEE JSAC*, Vol. 18(3), March 2000.
[17] J. Ha and C. Choi, "*Dynamic Optimization of IEEE 802.11 CSMA/CA Based on the Number of Competing Stations*", Globecom 2006.
[18] S.-J. Han, et.al., "*Analysis of Spatial Unfairness in Wireless LANs*", in *IEEE INFOCOM 2009*, Brazil, March 2009.