Analysis of Collaborative Distributed Admission Control in 802.11x Networks

Thinh Nguyen, *Member, IEEE,* Kien Nguyen, *Member, IEEE*, Linhai He, *Member, IEEE,*

Abstract

With the recent surge of wireless home networks, it is increasingly important to employ admission control mechanisms in order to enhance the performance of the wireless multimedia applications. In this paper, we extended the work in [1] for performing distributed admission control in a collaborative wireless environment. In particular, we provide an in-depth analysis on the throughput and jitter of traffic of the proposed admission control algorithms for collaborative wireless networks. Simulation results confirm our theoretical predictions on the performance of the proposed admission control algorithms.

I. Introduction

Recent years have witnessed an explosive growth in multimedia wireless applications such as video streaming and conferencing. One of the reasons for this tremendous growth is the deployment of IEEE 802.11x WLANs [2] in both private homes and enterprise networks. Without a proper bandwidth allocation mechanism in a wireless network, a flow may experience a significant performance degradation due to a large number of flows. To alleviate this problem, the recent 802.11e standard [3][4][5] specifies a mechanism for traffic differentiation. However, traffic differentiation alone cannot guarantee the performances for individual flows sharing the wireless medium. Instead, admission control mechanism is needed to ensure that a new flow is admitted, only if there is enough resource to support the required throughputs of all the existing flows.

Unlike the admission control in a wired network, admission control for a wireless network is more complex. This complication is due to the channel contention access, in which the interferences i.e., the collisions between the new flow and the existing flows reduce all the flow's throughputs. The number of these collisions increases nonlinearly with the number of competing flows, making it harder for an admission control algorithm to decide whether or not to admit a new flow. Many existing admission control algorithms for 802.11x based networks have been proposed [6][7][8]. Most of them assume an Access Point (AP) that runs the admission control algorithm. The advantage of this approach is that the AP has complete information about the network, and thus can allocate the resource efficiently. For example, Xiao and Li [6][9] uses the measurements to provide flow protection (isolation) in 802.11e network. The algorithm requires the AP to broadcast the necessary information to other wireless stations.

On the other hand, centralized admission control is not possible when multiple wireless devices form an ad-hoc network. Thus, the devices must collaborate with each other to jointly satisfy their bandwidth requirements. In this paper, we describe and analyze the performance of a collaborative distributed admission control (CDAC) algorithm in which, every device are assumed to be honest and collaborative [1]. Specifically, a device will not start a new flow if by doing so, there is not sufficient resources to support the existing flows. Our discussion will be limited to a one-hop wireless network, e.g. the network of all the wireless devices within a home or a small building such

Thinh Nguyen and Kien Nguyen are with the Oregon State University.

Linhai He is with Boston Consulting Group.

that, every device can hear the transmissions of all other devices. We first discuss the background and a few related work in Section II. Next, we describe the proposed CDAC algorithm and its analysis in Section III. Section IV is devoted to the discussion of the simulation results. Finally, we summarize our paper in Section V.

II. Preliminaries

A. Contention Based Access in 802.11x Networks

Contention based access enables multiple devices to compete for a shared wireless channel. While there are many parameters in 802.11x standards, for simplicity, we focus our discussion on the contention window (CW) and $TXOP$. To access the channel, a device first senses the channel. If the channel is idle for more than the Arbitration Interframe Space (AIFS) time, the device starts sending the data. Otherwise, it sets a back-off timer for a random number of time slots between $[0, CW_{min}]$ where CW_{min} is the minimum contention window size. The backoff timer is decremented by one for each idle time slot after the AIFS time, and halts decrementing when a transmission is detected. The decrementing resumes when the channel is sensed idle again for an AIFS time. A device can begin transmission on the channel as soon as its backoff timer reaches zero. If a collision occurs, i.e., no acknowledgment packet is received after a short period of time, the back-off timer is chosen randomly between $[0, (CW_{min}+1)2^i - 1]$ where i is the number of retransmission attempts. Clearly, a flow uses a small CW_{min} is likely to obtain the channel than a flow that uses a large CW_{min}.

One problem with contention based access in the 802.11 WLAN is the *nonlinear* throughput reduction of the existing flows with the introduction of a new flow due to higher collision rate. To that end, Pong and Moors [10] devises an algorithm for computing the new $CW'_{min}s$ and $TXOP$'s for the existing flows to maintain their required throughputs. Their algorithm is based on the analytical model proposed by Bianchi [11]. In this model, the transmission probability $p_t(i)$ of a flow i can be derived with the following formula:

$$p_t(i) = \frac{2(1-2p_c(i))}{(1-2p_c(i))(W+1) + p_c(i)W(1-(2p_c(i))^b)}, \tag{1}$$

Where $p_c(i)$ is the long term collision probability of flow i, W is the CW_{min} used for flow i, and b is the maximum backoff stage with $CW_{max} = (CW_{min}+1)2^b - 1$. Pong and Moors propose to monitor the collision probability $p_c(i)$ of each flow in order to calculate transmission probability p_t. Once p_t is obtained, one can compute the probability of a successful slot $p_s(i)$ for a flow i. The throughput of a flow i then can be computed using $p_s(i)$. Please see [10] for the detail calculations.

The PM algorithm makes an approximation by assuming that the collision probability of a flow i remains constant before and after the new flow joins. Another difficulty with the PM algorithm is that when a new flow requests to join, the AP has to approximate the collision rate of a new flow as the collision rate of an existing flow with similar throughput. If a flow with similar throughput is not available, the algorithm may produce inaccurate throughputs for all the flows.

III. Collaborative Distributed Admission Control (CDAC)

Our algorithm is similar to the PM algorithm in the sense that every device also adjusts its transmission parameters to maintain their required throughputs in presence of the new flow. However there are several major differences

as follows. First, unlike the PM algorithm, we do not assume that the collision probability of every flow remains constant before and after the new flow joins. Therefore, our algorithm can accurately calculate the throughput for each flow. Second, guessing the collision probability of a new flow is not required in our algorithm. Third, our algorithm is designed to be performed distributedly while the PM algorithm must be done at the AP. Finally, unlike the PM algorithm, our algorithm assumes that the contention window size is not doubled after every unsuccessful retransmission attempt. We argue that doubling of contention window size is not optimal when a proper admission control is used. Using admission control, the traffic load at any given time is known or highly predictable, and therefore the contention window size and/or the $TXOP$ of each device can be precisely tuned (by the admission control algorithm) to achieve the required throughput. Therefore, we expect higher performance from this approach than that of blindly doubling the contention window size after every unsuccessful retransmission attempt. Furthermore, we consider a small wireless network in which every wireless device can hear each other's transmissions such as a network of all the wireless devices (without an AP) within a home. We also assume the use of reservation packets RTS/CTS. In our proposed scheme, each wireless device continuously monitors traffic in its neighborhood and estimates the following parameters:

- I: percentage of idle slots (denoted as type-I slots).
- S_i: percentage of slots with a successful transmission of RTS/CTS packet (denoted as type-S slots) of flow i.
- C: percentage of slots with collision (denoted as type-C slots).
- D_i: throughput of flow i as a fraction of the channel capacity.

Note that $I + C + \sum_i S_i = 1$. Also, if admission control is performed at the AP, these parameters can be easily obtained.

Our analysis is based on time-slotted, reservation based protocols, where the time taken to make a reservation is a geometrically distributed random variable with parameter p. To translate the transmission probability p back to the contention window size used in IEEE 802.11x protocol, W can be set to $1/p$. We note that this is only an approximation since W in IEEE 802.11x protocols are not reset at every time slot.

Suppose a devices wants to start a new flow of throughput x kbps. It first needs to decide whether the network has sufficient resources to support the new flow. If it does, how would each device tune its transmission parameters, e.g. CW, to take into account of the new traffic in order to maintain the required throughputs. Each device is assumed to be collaborative such that it will not inject a new flow if there is not enough bandwidth for the existing flows. *It is important to note that all the admission control operations are done by the device that wants to inject a new flow. If and only if the new flow is admitted by the admission control algorithm running locally on the same device, then this device will broadcast the updated transmission parameters to other devices.* This approach reduces the workload for other devices. In this paper, we only consider tuning the contention window size.

A. Theoretical Performance of N flows

We assume that every device can start at most one flow at any point in time. A straightforward generalization to support multiple flows per device is to consider all the flows from one device as one single large flow with the transmission probability p. Whenever a device successfully obtains the channel, it selects a packet from one of its

flows to send. The probability of a packet selected from a particular flow then equals to the ratio of that flow's throughput to the total throughput of all the flows on the same device. This approach would result in the correct average required throughputs for all the individual flows.

We now show how to compute I, C, and S_i's, given the transmission probabilities p_i's. Suppose the transmission probability for a new flow is p, then for the *type-C* slots, in which collisions occur, the new traffic would have no impact on them. For the *type-S slots*, with probability p, it may cause a collision in that slot. For any *type-I* slot, with probability p, it would become a *type-S* slot. Otherwise it stays the same. Using the above argument, we can calculate the percentages of *C-type*, *I-type*, and *S-type* slots after the new flow starts. In particular, the new idle, collision, and successful slots can be calculated using the current I, C, S, and p as:

$$I_{new} = I_{current} - I_{current}p \tag{2}$$

$$C_{new} = C_{current} + S_{current}p \tag{3}$$

$$S_{new} = S_{current}(1-p) + I_{current}p. \tag{4}$$

Here, we denote $S = \sum_i S_i$. Similarly, we can calculate the percentages of successful slots S_i as

$$S_{i,new} = S_{i,current}(1-p), \tag{5}$$

for any existing flow i, and the percentage of successful slots for the new flow as

$$S_N = I_{current}p. \tag{6}$$

Using the equations above, it is straightforward to derive an algorithm for computing I, C, S_i's for N users in terms of the transmission probabilities $p_1, p_2, ...p_N$ iteratively. Due to limited space, we omit the algorithm's listing and simply call this algorithm: Algorithm 1.

In principle, the Algorithm 1 for computing S_i's in terms of p_i's produces a set of N equations with N unknown variables p_1, p_2, ... p_N. Hence, one can solve for p_i's based on S_i's. Unfortunately, these equations are not linear, and therefore difficult to solve. We propose an algorithm to find the p_i's given S_i's based on the following observation.

We note that when a flow i stops, I will increase by S_i. If flows i starts again with the same transmission probability p_i as before, its percentage of successful slots remains S_i as before. Hence, the following equations hold:

$$(I + S_i)p_i = S_i; p_i = \frac{S_i}{I + S_i}. \tag{7}$$

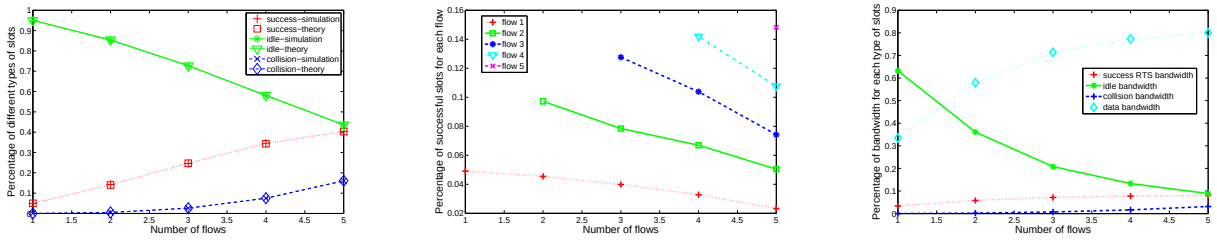
This is true because $I + S_i$ is the percentage of idle slots without flow i. Hence, after the flow i starts, its percentage of successful slots is $(I + S_i)p_i$ which should also precisely equals to S_i, the successful percentage before it stops. Thus, we have N such equations corresponding to N flows. We also have another equation:

$$I + C + S = 1, \tag{8}$$

where C and S are the percentages of collision and successful slots for all the flows. We note that I is the same for every equation since it is the percentage of idle slots when all flows are active. Now, we can solve for $N + 1$ unknowns, i.e. N p_i's and I. Solving this set of $N + 1$ equations is simple since each equation is linear except

Equation (8). However, Equation (8) will be used as a constraint. The idea is to systematically try different values of I from small to large, e.g. 0 to 1. For each value of I, we compute p_i's according to Equation (7). All the p_i's are then input the Algorithm 1 to compute C and S. We then test to see whether or not $I + C + S$ approximately equals to 1. If so, we have an admissible set of solutions. If not, we increase I by a small value and repeat the procedure. If the algorithm cannot find satisfied p_i's for the entire range of $I \in [0, 1]$, then this implies invalid S_i's. Due to limited space, we omit the algorithm's listing, and simply call this algorithm: Algorithm 2.

To verify our theoretical results, we simulate the following scenario. Five flows start in the increasing order - with flows 1 and 5 being the first and the last in the network (note that the starting order of flows does not change the results). The transmission probabilities of flows 1 to 5 are 0.05, 0.1, 0.15, 0.2, and 0.25, respectively. The $TXOP$ for all the flows is set to 10 time slots and remains constant throughout the simulation. Figure 1(a) shows the percentages of different slot types as a function of the number of flows for both theory and simulation. As seen, the theory and simulation results match each other almost exactly for different types of slots. As the number of flows increases, the collision and successful probabilities increases while the idle probability decreases. This result is plausible when the network load is still light. In other words, every flow succeeds in getting their slots from a large number of idle slots. Figure 1(b) shows the percentages of successful slots for different flows decreases as the number of flows increases as expected. Finally, Figure 1(c) shows the percentages of the total bandwidth taken by different types of slots as the number of flows increases.



(a) (b) (c)

Fig. 1. *Five flows joins the network sequentially, each with different transmission probabilities; (a) The percentages of successful, collision, and idle slots vs. the number of flows from theory and simulation; (b) Percentages of successful slots of all flows as the number of flows increases; (c) Percentage of bandwidth taken by different types of slots.*

B. Admission Control Algorithm

Assume a fixed $TXOP$, we proceed to calculate the throughput in terms of the number of time slots per unit time. The throughput in kbps can be easily obtained by multiplying the *time slot* throughput by the number of bits per time slot. In addition, we assume that every device knows the maximum capacity of the channel. Thus, all the calculations of throughput will be done in terms of a fraction of the maximum channel capacity. It is straightforward to compute S_i in terms of $TXOP$ and the fractional throughput D_i's - the requested throughput as a fraction of the channel capacity as:

$$S_i = \frac{D_i}{S_i(1 - \sum_{j=1}^{N} D_j)} \tag{9}$$

Next, using Algorithm 2, p_i's can be computed given S_i's. The admission control algorithm (shown below) will accept the new connection only if there exists a set of valid p_i's.

Algorithm 3: *Admission Control*

```
Ad_Ctrl_p(D[i], TXOP, N) {
  D = sum of all D[i]'s;
  for i = 1:1:N {
    S[i] = D[i]/(TXOP(1-D));
  }
  // computing p's.  If algorithm 2 cannot find a set of p_i, the value of success is zero.
  [p[], success]  = Algorithm_2(S[], N);
  if success = 0 {
    new flow is rejected.
  }
  else {
    admit the new flow.
  }
}
```

When a device requests to start a new flow, it uses the Algorithm 2 to determine whether or not a set of p_i's exists. If so, it broadcasts the updated p_i's to other devices, and starts sending its data with its transmission probability calculated from Algorithm 2. Other the devices can start transmitting their flow with the new $p_i's$. Otherwise, if no p_i's exist, the new flow is rejected.

C. Throughput Jitter Analysis

The proposed admission control algorithm guarantee that each flow will obtain its required throughput when it is averaged over a long period of time. However, the throughputs of the flows may fluctuate within a short period of time. These fluctuations in throughput prevent smooth playback for many audio and video streaming applications. Thus, we would like to analyze the throughput jitter for different flows within a short period of time. A device can estimate the throughput jitters of existing flows and use them as additional requirements to decide whether or not to inject its new flow.

We proceed to quantify the throughput jitter of a flow as the normalized standard of deviation of its fractional throughput within a number of time slots. Specifically, it is:

$$Stdev_n(X_i(T)) = \frac{Stdev(X_i(T))}{D_i}, \tag{10}$$

Where $X_i(T)$ is a fraction of the data slots of the flow i measured within T time slots, and D_i is its average long term fractional throughput. Clearly, if the throughput is measured within small window of time (number of time slots), its variance will increase. Now, the amount of data sent by a flow i within a number of time slots is proportional to the number of successful transmissions of the RTS/CTS within that period of time. Thus, we can calculate the variances of the number of successful transmission slots within a period of time. The probability of successful transmission for a flow i for a slot is S_i. The number of non-data slots in T time slots is $T(1-\sum_i^N D_i)$ where D_i is the fractional throughput of flow i. Thus the number of successful transmission slots Y_i within this period is binomially distributed with the parameters S_i and $T(1-\sum_i^N D_i)$. Therefore, its variance is:

$$Var(Y_i) = T(1-\sum_i^N D_i)S_i(1-S_i). \tag{11}$$

By definition,

$$X_i(T) = Y_i TXOP_i/T, \tag{12}$$

Therefore, the variance of $X_i(T)$ is:

$$Var(X_i(T)) = \frac{(1-\sum_i^N D_i)S_i(1-S_i)TXOP_i^2}{T}, \tag{13}$$

and the standard of deviation of $X_i(T)$ is

$$Stdev(X_i(T)) = TXOP_i\sqrt{\frac{(1-\sum_i^N D_i)S_i(1-S_i)}{T}}. \tag{14}$$

Finally, the normalized standard of deviation of the fractional throughput equals to:

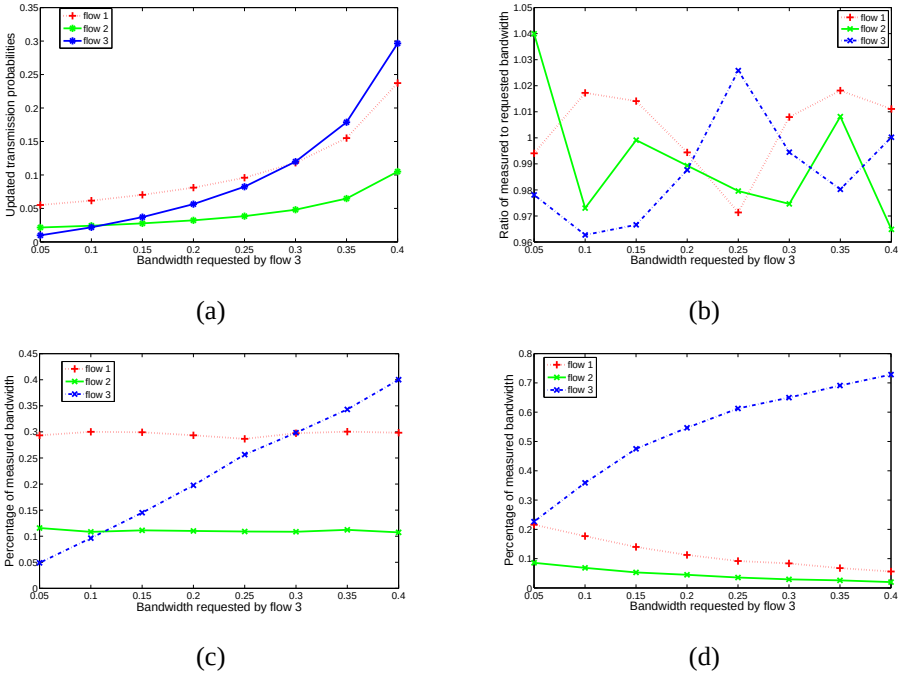
$$Stdev_n(X_i(T)) = TXOP_i\frac{\sqrt{(1-\sum_i^N D_i)S_i(1-S_i)}}{D_i\sqrt{T}} \tag{15}$$

IV. Simulation Results and Discussion

To evaluate the performance of our admission control algorithms, we set up the following simulation scenario. Two flows 1 and 2 are assumed to send data continuously while flow 3 requests to send data at different rates. Flow 3 uses the proposed admission control algorithm to find out whether or not there is enough bandwidth for it to send data. If so, it determines the parameters for the existing flows 1 and 2, and sends these parameters to all the flows. The transmission probabilities for flows 1 and 2 are set 0.05 and 0.02, respectively. The $TXOP$'s of all the flows are set to 10 time slots, and remain constant throughout the simulations. As a result, $D_1 = 0.29$ and $D_2 = 0.12$. Flow 3 requests different throughputs, ranging from 0.05 to 0.4 of the total capacity. Figure 2(a) shows the new transmission probabilities to maintain the required throughputs for all three flows. As flow 3 requests a larger portion of the throughput capacity, the transmission probabilities of all three flows increase to compensate for smaller probabilities of successfully obtaining the channel. Figure 2(b) shows the ratio of the measured throughput to the requested throughput for each flow. Figure 2(c) shows the corresponding measured throughputs. As seen, the algorithm is able to maintain the throughputs of the existing flows while provides the required throughput for the new flow. Finally, 2(d) shows the reduced throughputs of the flows if no admission control is used. We now show the throughput jitter for the proposed algorithm. The requested fractional throughputs D_i's are set to 0.25, 0.12, and 0.5, respectively. The $TXOP$ is set to 10 time slots. S_i's are then calculated in terms of $D_i's$ and fixed $TXOP$. The normalized standard of deviation of the fractional throughputs are then computed using Equation (15). Figure 3(a) shows the normalized standard of deviation as a function of the buffer size. As expected, as the buffer size increases, the normalized standards of deviation decreases. Figure 3(b) shows the normalized standard of deviations of throughputs as a function of the throughput of flow 3. As seen, the normalized stdev.'s of flows 1 and 2 remain constant while the normalized stdev's of flows 3 reduces as its throughput increases.

V. Conclusions

In summary, we extended the work in [1] for performing distributed admission control in a collaborative wireless environment. In particular, we provide an in-depth analysis on the throughput and jitter of traffic of the proposed admission control algorithms for collaborative wireless networks. Simulation results confirm our theoretical predictions on the performance of the proposed admission control algorithms.



cc

Fig. 2. *(a) New transmission probabilities as a function of flow 3's bandwidth; (b) ratio of measured to requested throughput as function of flow 3's bandwidth; (c) flow's fractional throughputs of flows 1 and 2 as a function of flow 3's bandwidth; (d) flow's fractional throughput without admission control as a function of throughput requested by flow 3.*

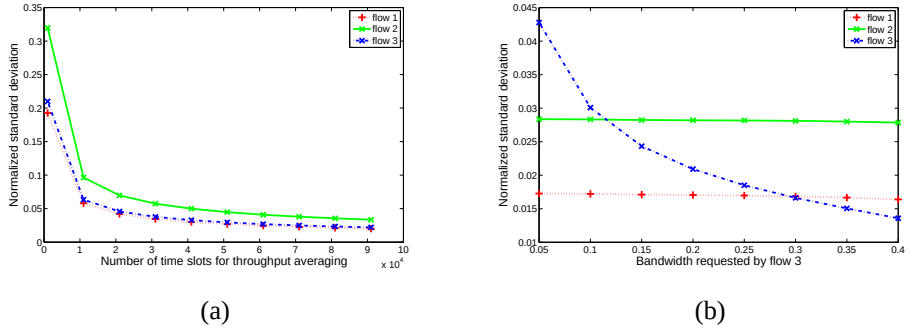


Fig. 3. *(a) Normalized standard of deviations as a function of (a) buffer size and (b) throughput of flow 3;*

REFERENCES

[1] T. Nguyen, K. Nguyen, and L. He, "Collaborative distributed admisison control for 802.11x networks," in *Submitted to ICC*, 2007.

[2] IEEE Std. 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999.

[3] X. Yang, "Ieee 802.11e qos provisioning at the mac layer," *IEEE Wireless Communication Magazine*, pp. 72–79, March 2004.

[4] IEEE Std. 802.11e/D6.0, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Quality of Service (QoS) Enhancements*, 2003.

[5] S. Mangold and et.al, "Analysis of ieee 802.11e for qos support in wireless lans," *IEEE Wireless Communication Magazine*, pp. 40–50, March 2003.

[6] Y. Xiao and H. Li, "Voice and video transmissions with global data parameter control for the ieee 802.11e enhanced distributed channel access," *IEEE Transactions on parallel and distributed systems*, vol. 15, Novemeber 2004.

[7] YD. Gao, J. Cai, and K. Ngan, "Admission control in ieee 802.11e wireless lans," *IEEE Network*, August 2005.

[8] D. Gu and J. Zhang, "A new measurement-based admission control method for ieee 802.11 wireless local area networks," *IEEE*, 2003.

[9] Y. Xiao, H. Li, and S. Choi, "Protection and guarantee for voice and video traffic in ieee 802.11e wireless lans," *IEEE*, 2004.

[10] D. Pong and T. Moors, "Call admission control for ieee 802.11 contention access mechanism," in *Globecom*, December 2003.

[11] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE JSAC*, pp. 535–57, March 2000.